Naval Research Laboratory

Stennis Space Center, MS 39529-5004

NRL/MR/7322--01-8266

Software Design Description for the Globally Relocatable Navy Tide/Atmosphere Modeling System (PCTides)

GRAEME D. HUBBERT

Global Environmental Modeling Service
P.O. Box 149 Warrandyte
Victoria, Australia

RUTH H. PRELLER
PAMELA G. POSEY

Ocean Dynamics and Prediction Branch
Oceanography Division

SUZANNE N. CARROLL

Planning Systems Incorporated
115 Christian Lane
Slidell, LA

April 23, 2001

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	April 23, 2001	Final

4. TITLE AND SUBTITLE

Software Design Description for the Globally Relocatable Navy Tide/Atmosphere Modeling System (PCTides)

5. FUNDING NUMBERS

6. AUTHOR(S)

Graeme D. Hubbert,* Ruth H. Preller, Pamela G. Posey, and Suzanne N. Carroll,**

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Research Laboratory
Stennis Space Center, MS 39529-5004

8. PERFORMING ORGANIZATION REPORT NUMBER

NRL/MR/7322--01-8266

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

SPAWAR
4301 Pacific Hwy., Code PWM185
San Diego, CA 92110-3127

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

*Global Environmental Modeling Services, P.O. Box 149 Warrandyte, Victoria, Australia
**Planning Systems Incorporated, 115 Christian Lane, Slidell, LA 70458

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

The purpose of this Software Design Description (SDD) is to describe the software design and code of the Globally Relocatable Navy Tide/Atmosphere Modeling System (PCTides). It includes the mathematical formulation and solution procedures for Global Coastal Ocean Model 2D/3D (GCOM2D/3D) and Mesoscale Atmospheric Prediction System (MAPS) as well as flow charts and descriptions of the programs, subprograms, and common blocks. PCTides is part of the Oceanographic and Atmospheric Master Library (OAML) and is actively configuration managed under the direction of that authority. This document, along with the Software Requirements Specification (Preller et al., 2001) and the Software Test Description (Preller et al., 2001) form the standard documentation package for the OAML PCTides.

14. SUBJECT TERMS

Tide model
Relocatable models
Tidal height prediction

15. NUMBER OF PAGES

104

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev 2-89)
Prescribed by ANSI Std 239-18
298-102

TABLE OF FIGURES

1.0 SCOPE

1.1 Identification

The Computer Software Configuration Item (CSCI), identified as the Globally Relocatable Navy Tide/Atmosphere Modeling System (PCTides) consists of a 2- and 3-dimensional barotropic tide/surge model, called the Global Environmental Modeling Services (GEMS) Coastal Ocean Model (GCOM2D and GCOM3D), and a Mesoscale Atmospheric Prediction System (MAPS).

GCOM2D is a depth-integrated shallow water model designed to characterize sea level and currents on or near continental shelves. It features a wetting and draining algorithm for simulating coastal flooding due to tides or storm surge.

GCOM3D is the three-dimensional counterpart to GCOM2D. It is a barotropic model for applications where current structure with vertical depth is required and tidal and wind forcing are dominant. Atmospheric forcing for GCOM2D/3D is provided by an existing operational Navy model, by the MAPS system, by an analytical hurricane vortex model, or by direct point observation.

MAPS is a hydrostatic primitive equations model designed to provide high-resolution representations of anemometer level winds and surface pressure as atmospheric boundary conditions for GCOM2D and GCOM3D. To this end, the turbulence closure scheme has been designed to allow the model to be run with its lowest model level at anemometer height, thus providing a direct simulation of the winds at this level. The equations of motion are coded in advective form and solved using a semi-implicit time differencing scheme ensuring that the model is both robust and economical to run, even in regions of steep terrain.

1.2 Document Overview

The purpose of this Software Design Description (SDD) is to describe the software design and code of the Globally Relocatable Navy Tide/Atmosphere Modeling System (PCTides). It includes the mathematical formulation and solution procedures for GCOM2D/3D and MAPS as well as flow charts and descriptions of the programs, subprograms, and common blocks. PCTides is part of the Oceanographic and Atmospheric Master Library (OAML), and is actively configuration managed under the direction of that authority. This document, along with the Software Requirements Specification and the Software Test Description form the standard documentation package for the OAML PCTides.

Manuscript approved February 23, 2001.

2.0 REFERENCED DOCUMENTS

2.1 Software Documentation Guidelines

Oceanographic and Atmospheric Master Library Summary. Naval Oceanographic Office, System Integration Department. OAML-SUM-21F. April, 1998.

Software Documentation Standards for Environmental System Product Development. Naval Oceanographic Office, System Integration Department. OAML-SDS-59A. January, 1999.

2.2 General Technical Documentation

Asselin, R.A., (1972): Frequency filter for time integrations. *Mon. Weather Rev.*, **100**: 487-490.

Blackadar, A.K., (1962): The vertical distribution of wind and turbulent exchange in a neutral atmosphere. *J. Geophys. Res.*, **67:** 3095-3102.

Charnock, H., (1955): Wind stress on the water surface. *Q. J. R. Meteorol. Soc.*, **81:** 639-640.

Dyer, A.J., (1974): A review of flux-profile relationships. *Bound. Layer Meteorol.*, **7:** 363-372.

ECMWF Research Department, (1984): Research Manual 3: ECMWF Forecast Model Physical Parameterization (J.-F. Louis, ed.), European Centre for Medium Range Weather Forecasts, Shinfield Park, U. K.

Galperin, B., L. Kantha, S. Hassid and A. Rosati, (1988): A quasi-equilibrium turbulent energy model for geophysical flows. *J. Atmos. Sci.*, **45:** 55-62.

Haltiner, J. and R.T. Williams, (1980): Numerical Weather Prediction and Dynamic Meteorology. John Wiley and Sons. 477pp.

Hammarstrand, U., (1977): On parameterization of convection for large scale numerical forecasts at mid-latitudes. *Contrib. Atmos. Phys.*, **50:** 78-88.

Hogan, T.F., Rosmond, T.E., and Gelaro, R., (1991). "The Description of the Navy Operational Global Atmospheric Prediction System's Forecast Model, " NOARL Report 13, Naval Research Laboratory, Stennis Space Center, MS.

Holland, G.J., (1980): An analytical model of the wind and pressure profiles in hurricanes. *Mon. Weather Rev.*, **108**: 1212-1218.

Hubbert, G.D., G.J. Holland, L.M. Leslie and M.J. Manton, (1991): A real-time system for forecasting tropical cyclone storm surges. *Weather and Forecasting*, **6**: 86-97.

Hubbert, G.D., L.M. Leslie and M.J. Manton, (1990): A storm surge model for the Australian region. *Q. J. R. Meteorol. Soc.*, **116:** 1005-1020.

Hubbert, G.D. and K.L. McInnes, (1999a): A storm surge inundation model for coastal planning and impact studies. *J. Coastal Res.*, **15**: 168-185.

Hubbert, G.D. and K.L. McInnes, (1999b): "Modeling Storm Surges and Coastal Ocean Flooding." In: Modeling Coastal Sea Processes. Editor B. J. Noye. World Scientific Publishing Co. p 159-188.

Kuo, H. L., (1965): On the formation and intensification of tropical cyclones through latent heat release by cumulus convection. *J. Atmos. Sci.*, **22**: 40-63.

Leendertsee, J. J., (1970): A water-quality simulation model for well-mixed estuaries and coastal seas: Vol. 1, Principles of Computation. The Rand Corporation, Santa Monica, Feb. 1970, RM-6230-RC.

Leslie, L. M., G. A. Mills, L. W. Logan, D. J. Gauntlett, G. A. Kelly, M. J. Manton, J. L. McGregor and J. M. Sardie, (1985): A high resolution primitive equations NWP model for operations and research. *Aust. Meteorol. Mag.*, **33:** 11-35.

McInnes, K. L. and G. D. Hess, (1992): The implementation of a high resolution boundary layer scheme in the ARPE model. *Aust. Meteorol. Mag.*, **40:** 21-31.

Mellor, G. L., and T. Yamada, (1974): A hierarchy of turbulence closure models for planetary boundary layers. *J. Atmos. Sci.*, **31:** 1791-1806.

Mellor, G. L., and T. Yamada, (1982): Development of a turbulence closure model for geophysical fluid problems. *Rev. Geophys. and Space Phys.*, **20:** 851-875.

Messinger, F., and A. Arakawa, (1976): Numerical methods used in atmospheric models. GARP Publication Series No. 14. WMO/ICSU Joint Organizing Committee, 64 pp.

Miller, M.J. and Pearce, R.P., (1974): Numerical model of a cumulonimbus. *Q. J. R. Meteorol. Soc.*, **100**: 133-154.

Miller, M. J. and Thorpe, A. J., (1981): Radiation conditions for the lateral boundaries of limited-area numerical models. *Q. J. R. Meteorol. Soc.*, **107:** 615-628.

Monin, A. S. and A. M. Obukhov, (1954): Basic laws of turbulent mixing in the ground layer of the atmosphere. *Akad. Nauk SSSR Geofiz. Inst. Tr.*, **151:** 163-187.

Shapiro, R., (1970): Smoothing, filtering and boundary effects. *Rev. Geophys. and Space Phys.*, **8:** 359-387.

Shum, C.K., Woodworth, P.L., Andersen, O.B., Egbert G.D., Francis, O., King, C., Klosko, S.M., Le Provost, C., Li, X., Molines, J.-M., Parke, M.E., Ray, R.D., Schlax, M.G., Stammer, D., Tierney, C.C., Vincent, P., and Wunsch, C.I., (1997) Accuracy Assessment of Recent Ocean Tide Models. *J. Geophys. Res.*, **102**: 25173-25194.

Signell, R.P. and Butman, B., (1992): Modeling tidal exchange and dispersion in Boston Harbor. *J. Geophys. Res.*, **97:** 15591-15606.

Smith, S.D. and Banke, E.G., (1975): Variation of the sea surface drag coefficient with wind speed. *Q. J. R. Meteorol. Soc.*, **101:** 665-673.

3.0 CSCI-WIDE DESIGN DECISION

The MAPS system uses a direct access file format where all model variables at all levels are stored at each given time. Refer to section 5.4a.

4.0 CSCI ARCHITECTURAL DESIGN

4.1 CSCI Components

a) PCTides can be divided into three main software units: **CSC MAPS, CSC GCOM2D**, and **CSC GCOM3D**. These are briefly described below along with commonly used library subroutines and data libraries required for smooth operation of PCTides.

b, c) CSC MAPS- This program consists of 23 subroutines and 6 common blocks, the bulk of which are used to provide high resolution spatial representation of the wind and pressure field for the GCOM ocean models.

CSC GCOM2D- This program consists of 21 subroutines and 6 common blocks that aid in modeling sea surface height and mean barotropic current structure in coastal regions due to tides and atmospheric forcing.

CSC GCOM3D – This three-dimensional z-coordinate hydrodynamic model consists of two subroutines and two common blocks that are used for coastal ocean modeling where barotropic current structure with depth is required. The remaining components of the model are GCOM2D subroutines and common blocks. This CSC can be run in either barotropic (no thermal and density variation) or baroclinic modes (temperature and salinity are model variables), but only the barotropic mode is supplied with this installation.

Standard Library Routines for PCTides programs:

Input/output routines –The seventeen subroutines fall into four file formats:

a) Direct access files-all model variables at all levels are stored at each given time (9 subroutines).
b) Sequential files- used for storage of output data (4 subroutines).
c) Time series files- used for storing station information in both the atmospheric and ocean models (2 subroutines).
d) Topography files-standard ASCII format used to write the topography and bathymetry of a model grid (2 subroutines).

Projection routines- In the PCTides system, both MAPS and GCOM2D/3D utilize a standard Cartesian grid projection system for the solution of the equations of motion. Four subroutines aid in producing this grid system.

Date/Time Routines- Four subroutines are used to convert between date and time, which are specified on input files, and an invariant reference time used by the model.

Filtering routines- Two subroutines are included in the model to reduce numerical errors and computational instabilities that may lead to a spurious growth of short waves that may mask the actual numerical simulation or, in severe cases, could cause catastrophic instability.

Miscellaneous MAPS routines- Four subroutines in the atmospheric prediction model pertain to the solution of semi-implicit equations.

d) The PCTides model is relatively new and has not yet been incorporated into the OAML software collection.

e) PCTides takes up about 360 MB of disk space in the PC Windows environment. The program requires the same amount of disk space for UNIX operating systems.

4.2 Concept of Execution

PCTides is a platform independent system that may be run in the PC Windows environment executed through an interactive menu or at the command prompt under DOS mode or the UNIX operating systems. The interactive menu allows for a logical structure by which the user can establish the required environment for carrying out a simulation with either GCOM2D or GCOM3D in any part of the world's oceans. Tides and/or winds can drive PCTides.

Execution of this CSCI begins by defining the model domain and generating a grid of the analysis region. A bathymetry grid is then created and tidal boundary conditions are derived for the region from global tidal files containing the amplitude and phase of the tidal constituent in time zone zero (Greenwich). Wind data are acquired from three possible sources: 1) NOGAPS or some higher resolution, 2) manual data entry, 3) hurricane model adaptations.

Prior to execution of the GCOM program, several parameters must be specified such as a wind flag, tide flag, run time and tidal start time, and output file time step, among others. In addition, station locations must also be specified before the model can be executed successfully. The "STATIONS.DAT" file may have up to 30 stations defined per model run.

Next, the type of model must be specified. Both GCOM2D and GCOM3D may be run using the same set-up data. The user would normally choose GCOM3D if near-surface currents were required. GCOM2D is faster and is used to predict sea levels as a result of tidal and/or hurricane forcing, as well as to simulate coastal region inundation due to tidal ebb and storm surge or flooding.

The user must specify output display options. Display options allow the plotting of spatial fields or time series of sea levels and ocean currents. Output stations, at which time series prediction of sea levels and currents are stored, may also be specified.

Assistance for the system may be obtained in the PC Windows menu where the menu procedures are also supported with help functions for all actions of choice. Help is accessed by pressing the "F1" key while the cursor is on the input box in question.
A diagram illustrating the basic logic underlying operation of the CSCI through PC Windows or the command prompt is shown in (Figure 4.2-1).

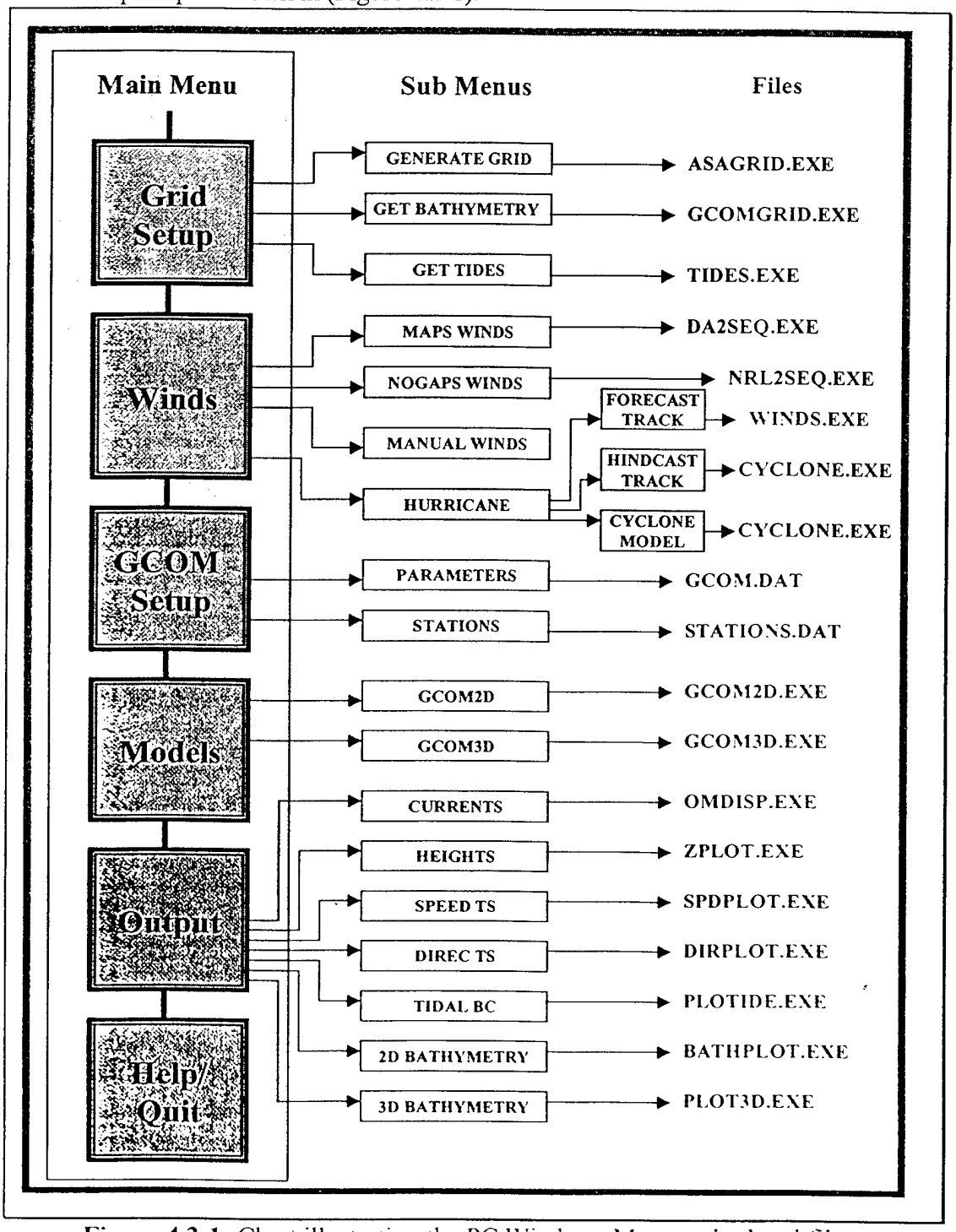

Figure 4.2-1: Chart illustrating the PC Windows Menu and related files.

4.3 INTERFACE DESIGN

4.3.1 INTERFACE IDENTIFICATION AND DIAGRAMS

The only Navy standard PCTides external interfaces are the input and output files.

The input files consist of the following types of data:
1. archived topographic data.
2. model domain parameters in 'GRIDGEN.DAT'.
3. Grenoble tidal data.
4. wind data from NOGAPS or some available higher resolution, from manual data entry, or derived from the hurricane model.
5. inputs to the coastal ocean model include various parameter and station location data.

The output files consist of the following types of data:
1. output stations where time series predictions of sea levels and currents are stored.
2. output display options consisting of sea level and current spatial field and time series plots.

The interfacing and operational environment for PCTides, which demonstrates the relationship between the components of the system and associated files, is illustrated in Figure 4.3-1.

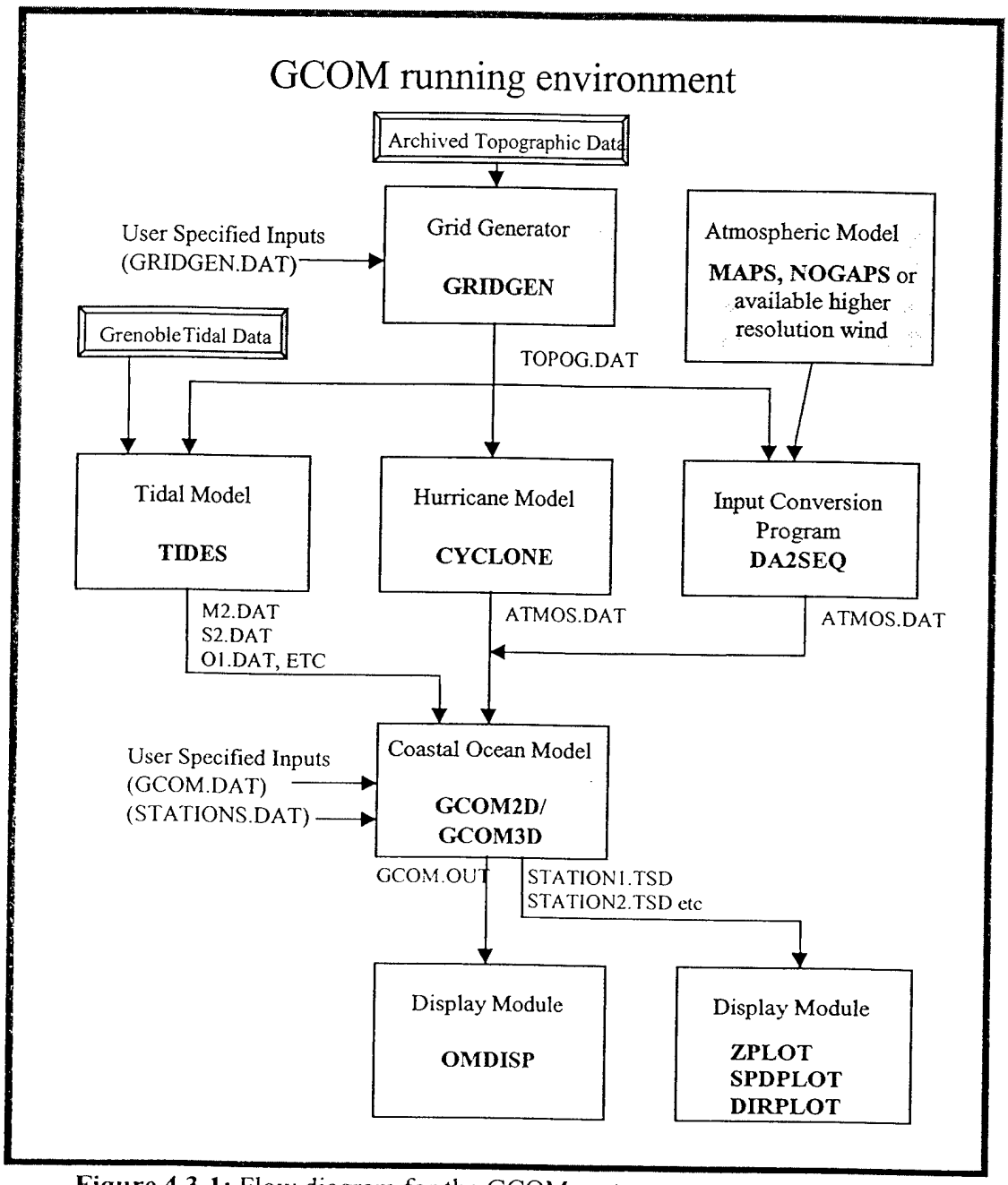

Figure 4.3-1: Flow diagram for the GCOM system.

5.0 CSCI DETAILED DESIGN

All routines are written in FORTRAN 90. The CSCI is platform independent and may be run in the PC Windows environment through use of a graphical user interface (GUI) interactive menu. It can also be operated through use of a command prompt in either UNIX or DOS mode.

The following paragraphs give a detailed description of the purpose, variables, logic, and constraints for the software elements in the CSCI. Description of the components of the program will be divided into design descriptions of the MAPS, GCOM2D, and GCOM3D models.

5.1 CSC Maps

MAPS is a grid-point, hydrostatic primitive equation model capable of being run on varying spatial resolution anywhere in the globe. The equations of motion are coded in advective form rather than flux form and solved using a semi-implicit time differencing scheme ensuring that the model is both robust and economical to run, even in regions of steep terrain. An important function of this model is to provide high-resolution representation of anemometer level winds and surface pressure as atmospheric boundary conditions for GCOM2D and GCOM3D. To this end, the turbulence closure scheme has been designed to allow the model to be run with its lowest model level at anemometer height, thus providing a direct simulation of the winds at this level. CSC Maps consists of 23 subroutines and 6 common blocks, which are interconnected to provide high resolution spatial representation of the wind and pressure field for the GCOM ocean models.

5.1.1 Constraints and Limitations

MAPS has been designed so that total grid arrays are not larger than 8100 points in the horizontal direction (for example, 90x90).

5.1.2 Logic and Basic Equations

CSC MAPS is based on the hydrostatic primitive equation model of Leslie et al., (1985). It differs however, in that the advective form of the primitive equation model is solved rather than the flux form. This is done to ensure greater computational stability of the model over regions of steep terrain without compromising model efficiency. The governing equations for momentum, mass, thermal energy and moisture, utilizing a normalized pressure or sigma coordinate in the vertical are:

$$\frac{\partial U}{\partial t} = -m^2\left(U\frac{\partial U}{\partial x} + V\frac{\partial U}{\partial y}\right) - \dot{\sigma}\frac{\partial U}{\partial \sigma} + fV - m^2\left(\frac{\partial \varphi}{\partial x} + RT\frac{\partial P_*}{\partial x}\right) + F_U + D_U, \quad (5.1.1)$$

$$\frac{\partial V}{\partial t} = -m^2\left(U\frac{\partial V}{\partial x} + V\frac{\partial V}{\partial y}\right) - \dot{\sigma}\frac{\partial V}{\partial \sigma} - fU - m^2\left(\frac{\partial \varphi}{\partial y} + RT\frac{\partial P_*}{\partial y}\right) + F_V + D_V, \quad (5.1.2)$$

$$\frac{\partial \varphi}{\partial \sigma} = -\frac{R_G T}{\sigma}, \quad (5.1.3)$$

$$\frac{\partial W}{\partial \sigma} = m^2\left(\frac{\partial U}{\partial x} + \frac{\partial V}{\partial y}\right) = -m^2\left(U\frac{\partial P_*}{\partial x} + V\frac{\partial P_*}{\partial y}\right) \quad (5.1.4)$$

$$\frac{\partial T}{\partial t} - \left(\frac{\alpha T}{\sigma} + \frac{\partial T}{\partial \sigma}\right)W - \sigma\frac{\partial T}{\partial \sigma}W_* = -m^2\left(U\frac{\partial T}{\partial x} + V\frac{\partial T}{dy}\right) + \alpha m^2 T\left(U\frac{\partial P_*}{\partial x} + V\frac{\partial P_*}{\partial y}\right) + H + F_T + D_T, \quad (5.1.5)$$

$$\frac{\partial R}{\partial t} = -m^2\left(U\frac{\partial R}{\partial x} + V\frac{\partial R}{\partial y}\right) - \dot{\sigma}\frac{\partial R}{\partial \sigma} + Q + F_R + D_R, \quad (5.1.6)$$

where U and V are the horizontal wind components in the direction of the map co-ordinates and are defined to be $U = \frac{u}{m}$ and $V = \frac{v}{m}$ where m is the map factor, $P_* = \ln p_*$ where p_* is the surface pressure and $W = \dot{\sigma} + \sigma W_*$ where $\sigma = \frac{p}{p_*}$, $\dot{\sigma}$ is the vertical velocity in σ-coordinates and $W_* = \frac{\partial P_*}{\partial t}$. The terms φ, T and R are the geopotential height, temperature and mixing ratio respectively; F_U, F_V, F_T and F_R are the turbulent exchange terms for momentum, heat and moisture; f is the Coriolis parameter, Q is a moisture source term and H represents adiabatic heating through radiation and latent heat release; D_U, D_V, D_T and D_R are the horizontal diffusion terms for momentum, heat and moisture and R_G and c_p are the gas constant and heat capacity at constant pressure for dry air.

5.1.2.1 Solution Procedure

The equations shown above are solved using a semi-implicit time-differencing scheme on a staggered Arakawa C-grid (Messinger and Arakawa, 1976) as shown in Figure 5.1-1. In the vertical, the terrain following σ-coordinates system is used as illustrated in Figure 5.1-2, with a top-down indexing system. The primary prognostic variables, U, V, T and R are defined on the 'full' model levels while $\dot{\sigma}$ is defined at the 'half' level.

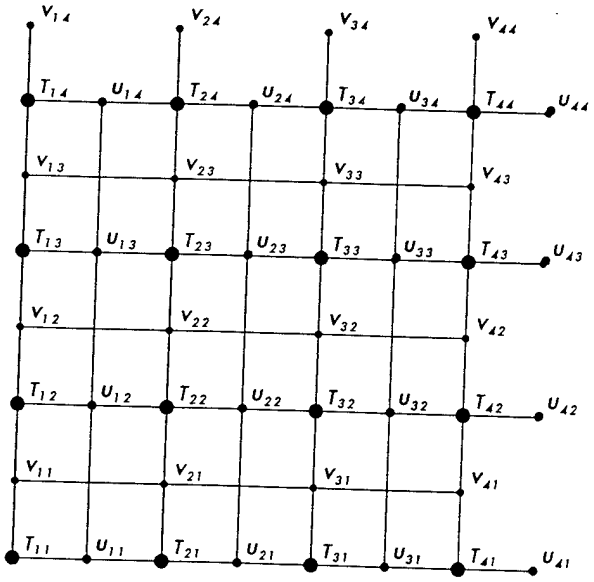

Figure 5.1-1: The horizontal grid structure used in MAPS.

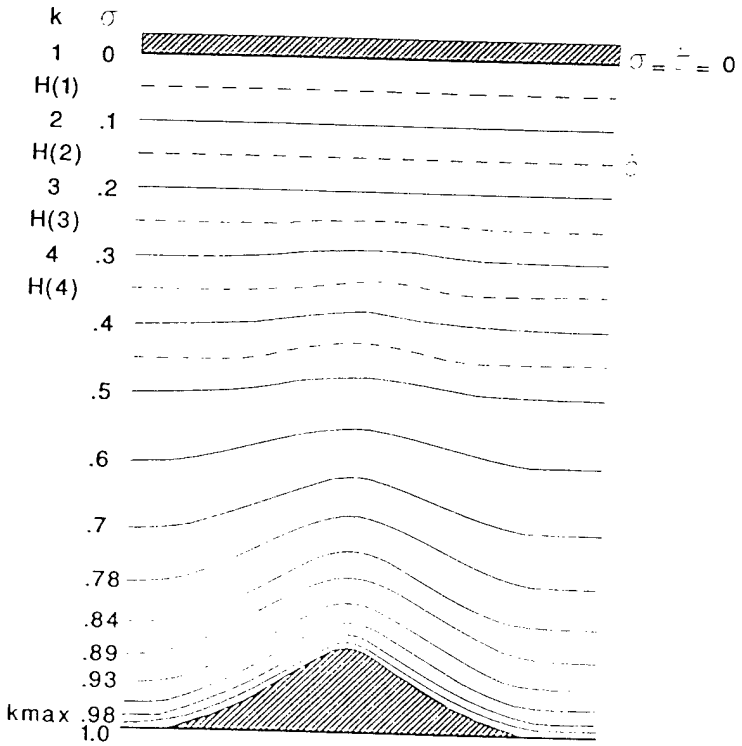

Figure 5.1-2: The vertical grid structure used in MAPS. All model variables are defined on the 'full' model level except for $\dot{\sigma}$.

By introducing a reference vertical temperature profile $T_o(\dot{\sigma})$ and defining averaging and differencing operators;

$$\overline{A}^{t} = \tfrac{1}{2}\left[A(t+\tfrac{1}{2}\Delta t) + A(t-\tfrac{1}{2}\Delta t)\right], \tag{5.1.7}$$

$$A^{t} = \left[A(t+\tfrac{1}{2}\Delta t) - A(t-\tfrac{1}{2}\Delta t)\right]/\Delta t, \tag{5.1.8}$$

equations (5.1.1) – (5.1.6) can be rewritten using semi-implicit differencing as

$$\overline{U}^{2t} + \Delta t \frac{\partial}{\partial x}\left((\overline{\varphi}^{2t} - \varphi_*) + R_G T_o\left[\overline{P}_*^{2t} - P_*(t-\Delta t)\right]\right) = U_{RHS} \tag{5.1.9}$$

where

$$U_{RHS} = U(t-\Delta t) - m^2 \Delta t\left(U\frac{\partial U}{\partial x} + V\frac{\partial U}{\partial y}\right) - \dot{\sigma}\frac{\partial U}{\partial \sigma} + fV - R_G(T_o - T)\frac{\partial P_*}{\partial x} + \frac{F_U}{m} + \frac{D_U}{m}, \tag{5.1.10}$$

$$\overline{V}^{2t} + \Delta t \frac{\partial}{\partial y}\left((\overline{\varphi}^{2t} - \varphi_*) + R_G T_o\left[\overline{P}_*^{2t} - P_*(t-\Delta t)\right]\right) = V_{RHS} \tag{5.1.11}$$

where

$$V_{RHS} = V(t-\Delta t) - m^2 \Delta t\left(U\frac{\partial V}{\partial x} + V\frac{\partial V}{\partial y}\right) - \dot{\sigma}\frac{\partial V}{\partial \sigma} - fU - R_G(T_o - T)\frac{\partial P_*}{\partial y} + \frac{F_V}{m} + \frac{D_V}{m}, \tag{5.1.12}$$

$$\varphi_k = \varphi_{k+1} + \frac{R_G T \Delta\sigma_k}{\sigma_k}(T_k + T_{k+1}), \tag{5.1.13}$$

$$\frac{\partial \overline{W}^{2t}}{\partial \sigma} + m^2\left(\frac{\partial \overline{U}^{2t}}{\partial x} + \frac{\partial \overline{V}^{2t}}{\partial y}\right) = -m^2\left(U\frac{\partial P_*}{\partial x} + V\frac{\partial P_*}{\partial y}\right) \tag{5.1.14}$$

$$\overline{T}^{2t} - \Delta t\left(\frac{\alpha T_o}{\sigma} - \frac{\partial T_o}{\partial \sigma}\right)\overline{W}^{2t} - \Delta t\sigma\frac{\partial T_o}{\partial \sigma}\overline{W}_*^{2t} = T_{RHS}, \tag{5.1.15}$$

where

$$T_{RHS} = T(t-\Delta t) - m^2 \Delta t\left(U\frac{\partial T}{\partial x} + V\frac{\partial T}{dy}\right) + \alpha m^2 T\left(U\frac{\partial P_*}{\partial x} + V\frac{\partial P_*}{\partial y}\right) \qquad (5.1.16)$$
$$+ \Delta t\left[\left(\frac{\alpha T}{\sigma} - \frac{\partial T}{\partial \sigma}\right)W - \left(\frac{\alpha T_o}{\sigma} - \frac{\partial T_o}{\partial \sigma}\right)W\right]$$
$$+ \Delta t\left[\sigma\frac{\partial T}{\partial \sigma}W_* - \sigma\frac{\partial T_o}{\partial \sigma}W_*\right] + \Delta t(H + F_T + D_T),$$

and

$$\overline{R}^{2t} = R(t-\Delta t) - m^2\Delta t\left(U\frac{\partial R}{\partial x} + V\frac{\partial R}{\partial y} - \dot{\sigma}\frac{\partial R}{\partial \sigma} + Q + F_R + D_R\right), \qquad (5.1.17)$$

From these equations, the hydrostatic equation is solved to provide geopotential heights. The thermodynamic equation is then transformed into a Helmholtz equation using the continuity and momentum equations and solved. Finally, the momentum, continuity and moisture equations are solved.

5.1.2.2 Initialization

The model is initialized by nesting within interpolated analysis and prognosis fields from a coarser grid model covering a large region (usually the globe). In this case the model is nested inside the U.S. Navy Operational Global Atmospheric Prediction System (NOGAPS) (Hogan et al., 1991) available at 0000, 0600, 1200 and 1800 UTC each day.

5.1.2.3 Boundary Conditions

The boundary conditions are applied using a nesting procedure that blends the externally prescribed fields with the model-simulated values over a number of grid-points into the computational domain. Absolute boundary conditions are applied to each model variable, ϕ, with decreasing intensity from the boundary to some specified number of grid-points (typically less than or equal to 10) within the domain, according to the following equation;

$$\phi = (1-\alpha)\phi_p + \alpha\phi_e, \qquad (5.1.18)$$

where ϕ_p is the model predicted value, ϕ_e is the externally applied value, provided by analyses in this case, and α varies from 1 at the boundary to zero, n_{max} points into the domain according to a cosine function

$$\alpha = 0.5(\cos\pi(1 - n/n_{max}) + 1), n = 1, n_{max}. \qquad (5.1.19)$$

The penultimate grid-points are then smoothed to dampen out gravity waves that will otherwise accumulate at the boundaries and lead to numerical instability.

The model is capable of being nested within itself to provide higher resolution simulations. Under these circumstances, the boundary conditions are applied the same way as described above where the ϕ_e are provided by a previously performed lower resolution run.

5.1.2.4 Temporal Filtering

A temporal filter is applied to all prognostic variables to reduce the amount of energy in waves with high frequencies. All variables are filtered according to

$$\hat{\phi}^t = \nu\phi^t + \frac{1}{2}(1-\nu)(\phi^{t+1} + \hat{\phi}^{t-1}), \tag{5.1.20}$$

(Asselin, 1972) where $\hat{\phi}$ is the filtered variable. The coefficient ν has a value of 0.8.

5.1.2.5 Parameterization of Sub-grid Scale Processes

Terms governing the effects of various processes such as turbulent vertical diffusion, radiation and precipitation are included in the prognostic equations for the horizontal wind components, temperature and moisture. Some of these processes can be calculated explicitly while others must be parameterized. In the model, the effects of these various processes are treated as a separate adjustment to the adiabatic semi-implicit solution of the equations of motion.

5.1.2.6 Turbulence Closure Scheme

This section describes the representation of the vertical transports via subgrid-scale eddies. These turbulent flux terms arise as dissipative terms in the equations of motion through Reynolds averaging considerations. In the U and V-momentum equations, the significant contribution comes from the vertical components of the stress tensor, $-\overline{\rho uw}$ and $-\overline{\rho vw}$, respectively, while in the thermodynamic and moisture equations the analogous terms $-\overline{\rho\theta w}$ and $-\overline{\rho rw}$ represent the vertical transfer of heat and moisture, respectively. These flux terms give rise to a closure problem that is generally overcome by parameterizing in terms of mean flow variables.

The turbulence closure scheme available in MAPS is a simplified second order closure scheme that is described by McInnes and Hess (1992). By definition, second order closure implies that the model carries prognostic equations for second order moments and closure of the system of equations is achieved by parameterizing or simplifying third order moment terms. However, Mellor and Yamada (1974) describe a hierarchy of second order closure models of varying levels of complexity ranging from the most complex "level 4" model to the least complex "level 2" model. In simplest terms, the higher the level, the greater the number of prognostic equations for the second order moments that are retained in the scheme.

In MAPS, the turbulence closure scheme is the so-called "level 2¼" model described in Galperin et al., (1988), which is similar in complexity to the "level 2½" model of Mellor and Yamada (1982). In this model, a prognostic equation is retained for the turbulent kinetic energy, q, only, while all other turbulent flux equations are subjected to various simplifications and

approximations to reduce them to a set of diagnostic equations that relate other turbulent fluxes and mean flow variables.

$$\overline{u^2} = \frac{q^2}{3}\left(1 - \frac{6A_1}{B_1}\right) - \frac{6A_1 l}{q}\frac{\partial U}{\partial z}\overline{uw}, \tag{5.1.21}$$

$$\overline{v^2} = \frac{q^2}{3}\left(1 - \frac{6A_1}{B_1}\right) - \frac{6A_1 l}{q}\frac{\partial V}{\partial z}\overline{vw}, \tag{5.1.22}$$

$$\overline{w^2} = \frac{q^2}{3}\left(1 - \frac{6A_1}{B_1}\right) - \frac{6A_1 l}{q}\beta g\overline{\theta w}, \tag{5.1.23}$$

$$\overline{uw} = -\frac{3A_1 l}{q}\left[(\overline{w^2} - C_1 q^2)\frac{\partial U}{\partial z} - \beta g\overline{u\theta}\right], \tag{5.1.24}$$

$$\overline{vw} = -\frac{3A_1 l}{q}\left[(\overline{w^2} - C_1 q^2)\frac{\partial U}{\partial z} - \beta g\overline{v\theta}\right], \tag{5.1.25}$$

$$\overline{uv} = -\frac{3A_1 l}{q}\left(\overline{uw}\frac{\partial V}{\partial z} + \overline{vw}\frac{\partial U}{\partial z}\right), \tag{5.1.26}$$

$$\overline{u\theta} = -\frac{3A_1 l}{q}\left(\overline{vw}\frac{\partial \Theta}{\partial z} + \overline{w\theta}\frac{\partial U}{\partial z}\right), \tag{5.1.27}$$

$$\overline{v\theta} = -\frac{3A_1 l}{q}\left(\overline{vw}\frac{\partial \Theta}{\partial z} + \overline{w\theta}\frac{\partial V}{\partial z}\right), \tag{5.1.28}$$

$$\overline{w\theta} = -\frac{3A_1 l}{q}\left(\overline{w^2}\frac{\partial \Theta}{\partial z} + \beta g\overline{\theta^2}\right), \tag{5.1.29}$$

$$\overline{\theta^2} = -\frac{B_2 l}{q}\overline{u\theta}\frac{\partial \Theta}{\partial z}. \tag{5.1.30}$$

The equations arise from the Reynolds Stress equations after scaling assumptions have been made and parameterizations introduced for many of the terms such as the triple correlation terms, pressure redistribution terms, dissipation and Coriolis terms. The constants, A_1, A_2, B_1, B_2, C_1, have been chosen by Mellor and Yamada (1974, 1982) on the basis of laboratory experiments in neutral stability conditions where the turbulent kinetic energy production terms are balanced by energy dissipation. The values they recommend are;

$$(A_1, A_2, B_1, B_2, C_1) = (.92,.74,16.6,10.1,.08). \tag{5.1.31}$$

The turbulent length scale l which forms the basis of the second order closure assumptions is determined using an algebraic expression suggested by Blackadar (1962) above the surface layer;

$$l = \left(\frac{kz}{\frac{kz}{\lambda} + 1}\right), \tag{5.1.32}$$

where k is von Kármán's constant (k=0.4) and z is the height. Mellor and Yamada (1974) specify λ according to;

$$\lambda = \frac{\alpha \int_0^\infty zq\,dz}{\int_0^\infty q\,dz}, \tag{5.1.33}$$

where α is an empirical constant.

These equations are used to solve for the vertical eddy flux terms $-\overline{uw}$ and $-\overline{\theta w}$ which in turn allow eddy coefficients to be specified using;

$$K_M = \frac{-\overline{uw}}{\left(\frac{\partial U}{\partial z}\right)} \equiv \frac{-\overline{vw}}{\left(\frac{\partial V}{\partial z}\right)}, \tag{5.1.34}$$

$$K_H = \frac{-\overline{\theta w}}{\left(\frac{\partial \Theta}{\partial z}\right)}. \tag{5.1.35}$$

These eddy coefficients are applicable at the center of each model layer. The consequent subgrid-scale source terms in the momentum and thermodynamic equations in the model are then simply obtained by the vertical divergence of these fluxes in the vertical diffusion equations which for the u-component of velocity takes the form;

$$\frac{\partial U}{\partial t} = \frac{\partial}{\partial z} K_M \frac{\partial U}{\partial z}. \tag{5.1.36}$$

The second order closure scheme is applied at all levels above the surface layer. The eddy coefficients that arise from the algebraic manipulation of equations (5.1.21-5.1.30) are;

$$K_M = \frac{lq\,A_1\left[1 - 3C_1 - \frac{6A_1}{B_1} - 3A_2 G_H\left[(B_2 - 3A_2)\left(1 - \frac{6A_1}{B_1}\right) - 3C_1(6A_1 + B_2)\right]\right]}{(1 - 9A_1A_2G_H)(1 - 3A_2G_H(6A_1 + B_2))} \tag{5.1.37}$$

$$K_M \frac{lq\,A_q\left(1 - \frac{6A_1}{B_1}\right)}{1 - 3A_2G_H(6A_1 + B_2)} \tag{5.1.38}$$

where $G_H \equiv -\left(\frac{l}{q}\right)^2 \left(\frac{g}{\theta}\right)\frac{\partial \Theta}{\partial z}$.

The prognostic equation for the turbulent kinetic energy is;

$$\frac{\partial}{\partial t}\frac{q^2}{2}=\frac{\partial}{\partial z}\left(lqS_q\frac{\partial}{\partial z}\frac{q^2}{2}+\frac{3}{4}\left(.4B_3l\frac{w_*}{q}\right)\frac{q}{\Theta_v}K_H\left(\partial\theta_v\partial z-\gamma_{cg}\right)\right) \qquad (5.1.39)$$
$$-\overline{uw}\frac{\partial U}{\partial z}-\overline{vw}\frac{\partial V}{\partial z}+\frac{g}{\Theta_v}\overline{w\theta_v}-\frac{q^3}{B_1l},$$

where the constants S_q and B_3 are 0.2 and 0.3 respectively, $\gamma_{cg}=5\left(\frac{H_s}{\rho C_p w_* h}\right)$ is the counter gradient heat flux, and W_* is the convective velocity defined as follows;

$$W_*=\begin{cases}u_*\left(\frac{-h}{kL}\right)^{\frac{1}{3}}, & \text{in convective conditions;}\\ 0, & \text{otherwise.}\end{cases} \qquad (5.1.40)$$

The vertical diffusion equation is solved for the horizontal wind components, temperature and mixing ratio using an implicit numerical finite difference scheme which is given below for U;

$$\frac{U^{n+1}-U^n}{\Delta t}=K_M\left(\frac{U_{k+1}^{n+1}-2U_k^{n+1}+U_{k-1}^{n+1}}{2\Delta z}\right). \qquad (5.1.41)$$

5.1.2.7 *Monin-Obukhov Surface Layer*

The surface (constant flux) layer is treated using Monin-Obukhov similarity theory (Monin and Obukhov, 1954). This layer, which is typically about 10 m deep, is assumed to lie between the surface and the lowest model layer. Here, the fluxes for momentum, heat and moisture obey the following relations;

$$\frac{\partial W}{\partial z}=\frac{u_*}{kz}\Phi_M\left(\frac{z}{L}\right), \qquad (5.1.42)$$

$$\frac{\partial\Theta}{\partial z}=\frac{H}{\rho c_p u_* kz}\Phi_H\left(\frac{z}{L}\right), \qquad (5.1.43)$$

$$\frac{\partial R}{\partial z}=\frac{E_s}{\rho u_* kz}\Phi_H\left(\frac{z}{L}\right), \qquad (5.1.44)$$

where $W=(U^2+V^2)^{\frac{1}{2}}$, c_p is the specific heat of dry air, $u_*\equiv\sqrt{\frac{|\tau_s|}{\rho}}$ is the roughness length, ρ is the air density and L is the Monin-Obukhov length defined as;

$$L=\frac{\Theta_v u_*^3}{gk\frac{H}{\rho c_p}} \qquad (5.1.45)$$

where Θ_v is the virtual potential temperature, H is the sensible heat flux. The eddy viscosity coefficients are defined according to;

$$K_M = u_* l \Phi_M\left(\frac{z}{L}\right), \tag{5.1.46}$$

$$K_H = u_* l \Phi_H\left(\frac{z}{L}\right). \tag{5.1.47}$$

According to Dyer (1974), the following stability functions are found to provide the most accurate flux-gradient description; in the unstable case $\left(\frac{z}{L}\right) < 0$,

$$\Phi_H = \left(1 - 16\left(\frac{z}{L}\right)\right)^{-\frac{1}{2}}, \tag{5.1.48}$$

$$\Phi_M = \left(1 - 16\left(\frac{z}{L}\right)\right)^{-\frac{1}{4}}, \tag{5.1.49}$$

and for the stable case $\left(\frac{z}{L} > 0\right)$

$$\Phi_H = \Phi_M = \left(1 + 5\left(\frac{z}{L}\right)\right). \tag{5.1.50}$$

In neutral conditions, $\Phi_H = \Phi_M = 1$ giving rise to the well known logarithmic wind profile in equation (5.1.42). Equations (5.1.48) and (5.1.49) can be integrated analytically to give;

$$U_{k\max} = V_{k\max} = \frac{u_*}{k}\left[\ln\left(\frac{z_{k\max}}{z_0}\right) - \Psi_M\left(\frac{z_{k\max}}{L}\right) + \Psi_M\left(\frac{z_0}{L}\right)\right] \tag{5.1.51}$$

and

$$\Theta_{k\max} - \Theta_s = \left(\frac{H}{\rho c_p u_* k}\right)\left[\ln\left(\frac{z_{k\max}}{z_T}\right) - \Psi_M\left(\frac{z_{k\max}}{L}\right) + \Psi_M\left(\frac{z_T}{L}\right)\right], \tag{5.1.52}$$

where

$$\Psi_M(\zeta) = \begin{cases} 2\ln\left(\frac{1+\Phi_M^{-1}}{2}\right) + \ln\left(\frac{1+\Phi_M^{-2}}{2}\right) - 2\tan^{-1}\Phi_M^{-1}, & \zeta < 0; \\ 0, & \zeta = 0; \\ -5\zeta, & \zeta > 0. \end{cases} \tag{5.1.53}$$

and

$$\Psi_H(\zeta) = \begin{cases} 2\ln\left(\frac{1+\Phi_H^{-1}}{2}\right), & \zeta < 0; \\ 0, & \zeta = 0; \\ -5\zeta, & \zeta > 0, \end{cases} \tag{5.1.54}$$

The roughness length z_0, over the land is assumed (following common practice) to be a physical property of the underlying surface. Over the ocean, z_0 is specified using the formulation of Charnock (1955) where the roughness length is a function of the wind speed;

$$z_0 = .032\frac{|\tau_s|}{\rho g}. \tag{5.1.55}$$

The analogous term in the temperature equation, z_T, is related to the roughness length according to $z_T = .1\ z_0$.

5.1.2.8 Surface Temperature

Surface temperature over land masses is calculated using a 3-layer slab model based on the scheme used in the European Center for Medium Range Weather Forecasting (ECMWF) model, (ECMWF, 1984). The model is illustrated in Fig. 5.1-3 where T_S is the surface soil temperature assumed to vary on a diurnal time scale, T_D is the deep soil layer temperature assumed to vary on a time scale of several days. T_{CL} is assumed to be sufficiently deep that the temperature varies on climatological time scales and as such can be held constant. The depths of the layers are defined in cm as follows;

$$(D_1, D_2, D_3) = (7.2, 43.2, 43.2). \tag{5.1.56}$$

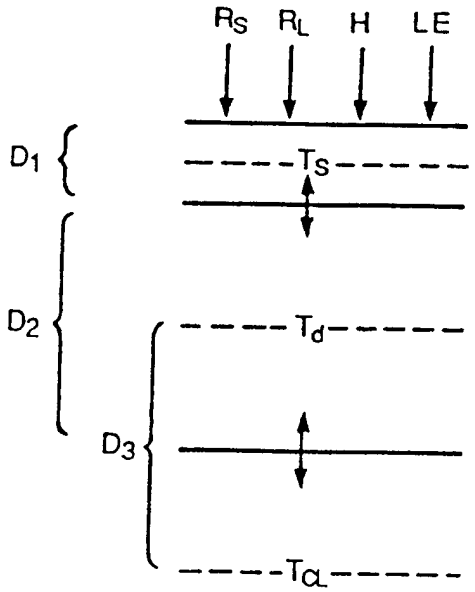

Figure 5.1-3: Schematic diagram showing the 3-layer surface temperature prediction scheme. The quantities R_S, R_L, H and LE represent the incoming shortwave and longwave radiation, sensible and latent heat fluxes, respectively. T_S and T_D are predicted mean temperatures over depths D_1 and D_2 respectively and T_{CL} is held constant. The thickness of the depths D_1, D_2 and D_3 are 7.2, 43.2, and 43.2 cm, respectively.

The equations for the surface and deep soil layers are;

$$\frac{\partial T_s}{\partial t} = \frac{\sum F}{\rho_g c_g D_1} + \frac{(T_d - T_s)\kappa}{\frac{1}{2} D_1 (D_1 + D_2)}, \tag{5.1.57}$$

$$\frac{\partial T_d}{\partial t} = \frac{(T_s - T_d)\kappa}{\frac{1}{2} D_1 (D_1 + D_2)} + \frac{(T_{Cl} - T_d)\kappa}{D_2 D_3}, \tag{5.1.58}$$

where ρ_g is the soil density, C_g is the specific heat capacity of the soil and κ is the thermal diffusivity. The transfer of heat through the soil is assumed to be through diffusive processes except in the surface layer where the first term on the RHS of equation (5.1.57) represents radiative and convective transfer of temperature with the atmosphere;

$$\sum F = R_S + R_L + E_S + H, \tag{5.1.59}$$

and R_S is the short wave solar radiation, R_L is the long wave radiation and E_S and H are the latent and sensible heat respectively defined according to;

$$H = \rho c_p C_H |\mathbf{U}_{k\max}| (T_{k\max} - T_s), \tag{5.1.60}$$

and

$$E_s = \rho L_E C_H \alpha |\mathbf{U}_{k\max}| (q_{sat} - q_{k\max}), \tag{5.1.61}$$

where C_H is the heat transfer coefficient, L_E is the latent heat and α is an efficiency coefficient defined as $\min(1, W_s / W_{ssat})$, where W_s is the soil moisture and W_{ssat} is the value of W_s when saturation occurs. A simplified surface radiation scheme is used to incorporate the effects of the radiative flux on surface temperature variation. This scheme provides the shortwave radiation in cloudy and clear sky conditions and a long wave radiation budget for frequencies between 800 and 1200 s^{-1}. Complete cooling to space is assumed for all frequencies outside this range.

Equation 5.1.57 is solved using the following finite difference procedure;

$$\frac{T_s^{n+1} - T_s^n}{\Delta t} = \phi(T_s)^{n+\frac{1}{2}}, \tag{5.1.62}$$

where $\phi(T_s)$ represents the right hand side of equation 5.1.57. By expanding this term in a Taylor series about $\phi(T_s)^n$ this equation becomes

$$\frac{T_s^{n+1} - T_s^n}{\Delta t} = \phi(T_s)^n + \frac{\Delta t}{2} \frac{\partial \phi}{\partial t}\bigg|^n, \tag{5.1.63}$$

which simplifies to the following explicit relationship for T_s^{n+1};

$$T_s^{n+1} - T_s^n = \frac{\Delta t \phi^n}{\left(1 - \Delta t \frac{\partial \phi^n}{\partial T_s}\right)}. \tag{5.1.64}$$

This 'Euler Backward' procedure is required only for T_s^{n+1} since once this is known, T_d^{n+1} can be solved explicitly.

5.1.2.9 *The Hydrological Cycle*

A 3-layer sub-surface model (Figure 5.1-4) analogous to the scheme used for surface temperature prediction is used for the calculation of surface wetness. The governing equations are;

$$\frac{\partial W_s}{\partial t} = -\frac{E}{\rho_{H_2O}} + P_R + \frac{(W_d - W_s)\lambda}{\frac{1}{2}D_1(D_1 + D_2)}, \tag{5.1.65}$$

$$\frac{\partial W_d}{\partial t} = \frac{(W_s - W_d)\lambda}{\frac{1}{2}D_2(D_1 + D_2)} + \frac{(W_{cl} - W_d)\lambda}{D_2 D_3}, \tag{5.1.66}$$

where E and P_R represent the moisture transfer to the ground from water vapor and precipitation respectively and ρ_{H_2O} and λ are the density and diffusivity of water. The values of D_1, D_2 and D_3 are the same as those used for the surface temperature prediction scheme and the moisture in the lowest sub-surface layer W_{Cl} is specified using climatological values.

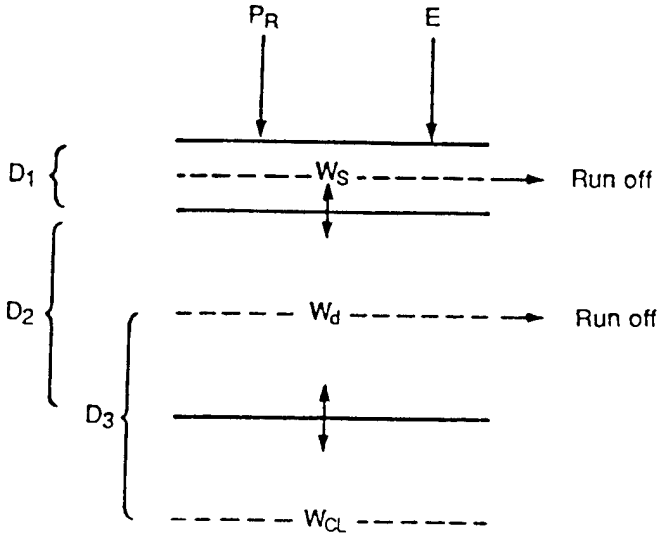

Figure 5.1-4: Schematic showing the 3-layer moisture scheme used by the model. The quantities E and PR represent the evaporation and precipitation respectively, while, W_s and W_d are the predicted moisture content over the depths D_1 and D_2.

5.1.2.10 Large Scale Precipitation

Large-scale condensation occurs in the model when the mixing ratio R exceeds a specified fraction of the saturated mixing ratio R_{sat}. The condensation rate is given by

$$C = \frac{R^* - R_{sat}}{2\Delta t(1 + L^2 q_{sat} / c_p R_G T^{*2})}. \quad (5.1.67)$$

Where T^* and R^* are the predicted values of temperature and mixing ratio before condensation. Due to the fact that latent heat release during condensation increases both temperature and R_{sat}, not all of the excess water condenses. The final values of R and T are given by;

$$R = R^* - 2\Delta t C \quad (5.1.68)$$

$$T = T^* + \frac{1}{c_p} 2\Delta t C \quad (5.1.69)$$

Condensed water is assumed to fall out immediately with no subsequent evaporation occurring. The precipitation rate due to large-scale rain is then given by;

$$P_{LS} = -\frac{1}{\rho_w g} \sum_{k=1}^{k\max} C_k P^* \Delta\sigma_k \quad (5.1.70)$$

where C_k is the condensation rate at level k.

5.1.2.11 Cumulus Convection

Subgrid-scale cumulus convection is parameterized using a scheme based on Kuo (1965) with modifications suggested by Hammarstrand (1977). Cumulus convection is assumed to occur when low level moisture convergence occurs in conjunction with an upper layer of conditionally unstable air. The extent of the cloud is taken to be the region bounded by the lifting condensation level of air near the surface and the intersection of the moist adiabat with the environmental sounding at upper levels. Cumulus clouds formed in this way are assumed to mix instantaneously and completely with the environment. The moistening and heating rates are proportional to the differences between the cloud temperature, T_C and mixing ratio, R_C and the environmental values T and R.

The cumulus convection scheme is divided into several steps:

(i) Check the sign of the large-scale moisture convergence, M_{conv}, where

$$M_{conv} = -\frac{P^*}{g}\int_{\sigma_T}^{\sigma_B}\left(\frac{\partial UR}{\partial x}+\frac{\partial VR}{\partial y}+\frac{\partial \dot{\sigma} R}{\partial z}\right)d\sigma \qquad (5.1.71)$$

If $M_{conv}>0$ then proceed to next step.

(ii) Check for conditional instability in the model vertical profile. That is, if

$$-\frac{\partial}{\partial\sigma}(\varphi + c_p T) > 0 \qquad (5.1.72)$$

$$-\frac{\partial}{\partial\sigma}(\varphi + c_p T + LR) < 0 \qquad (5.1.73)$$

(iii) Calculate cloud base by carrying out a dry adiabatic ascent of air starting from the lowest model level.

(iv) Calculate the cloud temperature profile $T_c(\sigma)$ from the equation for the moist adiabat;

$$\frac{\partial T_c}{\partial\sigma} = \frac{RT_c}{c_p\sigma}\left(1+\frac{Lq_s}{RT_c}\right)\Bigg/\left(1+\frac{L^2 q_s}{R_v c_p T_c^2}\right) \qquad (5.1.74)$$

Where R_v is the gas constant for water vapor. This equation is integrated using a Runge-Kutta method and is halted when cloud top is reached.

(v) The heating and moistening at level k are assumed to be proportional to the differences between the model cloud and the environment,

$$\Delta T_k = \xi_T (T_C - T) \qquad (5.1.75)$$

$$\Delta R_k = \xi_R (R_c - R) \qquad (5.1.76)$$

It is assumed that $\xi_R=\xi$ and $\xi_T=\beta\xi$ where β is a positive constant with value of 0.1. The value of ξ is calculated from

$$\xi = LM_{conv}/(\beta Q_1 + Q_2) \qquad (5.1.77)$$

where

$$Q_1 = \frac{P^*}{g}\int_{\sigma_1}^{\sigma_B} c_p (T_c - T)d\sigma \qquad (5.1.78)$$

$$Q_2 = \frac{P^*}{g}\int_{\sigma_1}^{\sigma_B} L(R_c - R)d\sigma \qquad (5.1.79)$$

Finally, the convective precipitation is computed from

$$P_C = \beta\xi\frac{P^*}{g}\int_{\sigma_1}^{\sigma_B}\frac{c_p}{L}(T_c - T)d\sigma \qquad (5.1.80)$$

5.1.2.12 Shallow Convection

A simple shallow convection scheme is employed up to level σ=0.7. This scheme was added to counter the tendency of the deep convection scheme to cause too much subsidence and moisture accumulation in the lowest model layers, particularly in the tropics. Shallow convection is assumed to occur if instability is observed in the levels above or below and the relative humidity of the current level is greater than 75%. In the event of shallow convection occurring, the eddy viscosity coefficients of momentum are increased to reflect the greater turbulence occurring within the level.

5.1.3 Data Elements

5.1.3.1 Commonly Used Parameters

File: amparam.for

Variable	Type	Description
nk	Integer	Maximum vertical dimension
ni	Integer	Maximum horizontal dimension (x-direction)
nj	Integer	Maximum horizontal dimension (y-direction)
nij	Integer	NI*NJ

These parameters are used in subroutines AMOUT, ASSLIN, BNDFIX2, CONST, CYCLE, FILTER, FILTER, FIZZIX, GETDT, PUTOUT, INNER, INNER2, NESTIN, PARAMS, SEMIMP.

File: amcommon.for

Variable	Type	Description
nkp1	Integer	NK+1
nout	integer	Number of output stations

5.1.3.2 Internally Defined Parameters

The following parameters are used in subroutine FIZZIX.

Variable	Type	Description
ya1,ya2,yb1, yb2,yb3,yc1	Real	Constants used in turbulence closure scheme.
za1,za2, za3, za4	Real	Constants used in solution of surface temperature and moisture equation.
yml	Real	Empirical constant, α, used in calculation of mixing length in turbulent diffusion scheme.
e1, e2, e3	Real	Constants used in solution of vertical diffusion equation.
cq	Real	Constant used in the calculation of θ_v, from θ and R.
emissg, emissc, stefbo	Real	Ground emissivity, cloud emissivity, and Stephan-Boltzman constant.
zr1, zt1	Real	Constants used in calculation of cloud transmissivity.
blimit, rhkuo	Real	Highest allowed cumulus base, critical relative humidity (Kuo scheme).
rv, hl	Real	Constants used in large-scale rain scheme.

The following parameters are used in subroutine NESTIN.

Variable	Type	Description
nbr, nbrp	Integer	Dimension for nesting fields, NBRP=NBR + 1.

Subroutines

5.1.3.3 Subroutine AMOUT

The purpose of this subroutine is to output model fields in direct access format. It is called by the MAIN program.

Calling sequence: subroutine amout(lu2,lu5,dt)

Data declarations: real dt
logical lu2,lu5

Arguments:

lu2	Logical unit number of the model run log file.
lu5	Logical unit number of the direct access file.
dt	Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.4 Subroutine ASSLIN

ASSLIN performs temporal filtering of U, V, T and RM arrays and is called by the main program.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

Algorithm:

Refer to Temporal Filtering, section 5.1.2.4.

5.1.3.5 Subroutine BNDFIX2

BNDFIX2 smoothes penultimate grid values of arrays PSP, UP, VP, RMP, TP, OMEGA, TSP, WSP, TDP and WDP after lateral boundary conditions are applied to prevent accumulation of gravity waves. Called by the MAIN program.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

Algorithm:

Refer to 5.1.2.3, Boundary Conditions.

5.1.3.6 Subroutine CONST

CONST sets up constants for horizontal diffusion, and eigenvectors and matrices for semi-implicit solution procedure. It is called by the MAIN program.

Calling sequences: subroutine const(dt)

Data Declarations: real dt

Arguments: dt Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.7 Subroutine CYCLE

CYCLE rotates time-stepping arrays at the end of each time step. It is called by the MAIN program.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.8 Subroutine DADADJ

Subroutine DADADJ performs dry adiabatic adjustment. It is called by SEMIMP.

Calling sequences: subroutine dadadj(dq, dqmqa5,tx, nz, ipc, jpc)

Data Declarations: real dq, dqmqa5, tx
integer nz, ipc, jpc

Arguments:		
	dq	array of $\Delta\ \sigma$.
	dqmqa5	Convective adjustment array.
	tx	Temperature calculated by DADADJ.
	nz	Vertical dimension.
	ipc	I-grid point (for diagnostic print).
	jpc	j-grid point (for diagnostic print).

5.1.3.9 Subroutine FILTER

FILTER performs spatial filtering of multi-level variables or their tendencies. Single level fields are filtered after every physics time step NPHYS. It is called by the MAIN program.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.10 Subroutine FIZZIX

FIZZIX calculates adjustments to U, V, T and RM due to vertical diffusion, cumulus convection and large-scale rainfall. Surface temperature and moisture changes are calculated. This subroutine is called by the MAIN program.

Calling sequences: subroutine fizzix(lu2, lu0, dt)

Data Declarations: real dt
logical lu2, lu0

Arguments:		
	lu2	Logical unit number of the model run log file.
	lu0	Logical unit number of the direct access file.
	dt	Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.11 Subroutine GETDT

GETDT evaluates the time-step based on the CFL condition and is called by the MAIN program.

Calling sequences: subroutine getdt(lu2, lu3, ntime, spmax, dt)

Data Declarations: real dt, spmax
integer ntime
logical lu2, lu3

Arguments:		
	lu2	Logical unit number of the model run log file.
	lu3	Logical unit number of the direct access file.
	ntime	Counter incremented by 1 each time a nesting file exists.
	spmax	Maximum wind speed found in the wind field array.
	dt	Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.12 Subroutine PUTOUT

PUTOUT outputs run-time information to the screen either in text format or graphically. It also writes a run-time log file (MAPS.LOG) and the stations time series file. It is called by the MAIN program.

Calling sequences: subroutine putout(llog, lutsd, lutsp, lu2,dt)

Data Declarations: real dt
logical llog, lutsd, lutsp, lu2

Arguments:		
	llog	Logical unit of MAPS.LOG.
	lutsd	Logical unit of the TSD files.
	lutsp	Inactive parameter.
	lu2	Inactive parameter.
	dt	Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.13 Subroutine INNER

INNER calculates the geopotential, calls INNER2, and calculates the forcing term for the solution of the Helmholtz equation. It is called by the MAIN program.

Calling sequences: subroutine inner(dt)

Data Declarations: real dt

Arguments: dt Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.14 Subroutine INNER2

INNER2 calculates the right hand side of the semi-implicit equations for U, V, T, and RM. It is called by subroutine INNER.

Calling sequences: subroutine inner2(dt)

Data Declarations: real dt

Arguments: dt Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.15 Subroutine NESTIN

NESTIN on initial call reads direct access header information, sets sigma levels and TBAR, initializes grid projection and calculates grid points nearest to specified stations for time series output. The weights are calculated for the nesting scheme, as well as solar declination and the top layer at which vertical diffusion is applied. Convective adjustment variables, constants for the semi-implicit scheme, and map factors at staggered grid points are defined. On subsequent calls, boundary conditions are applied to model predicted variables via a nesting technique. This subroutine is called by the MAIN program.

Calling sequences: subroutine nestin(lstn, llog, lu2, lu3, dt, iend)

Data Declarations: real dt
integer iend
logical lstn, llog, lu2, lu3

Argument:		
	lstn	Logical unit number of the model run log file.
	llog	Logical unit number of the model run log file.
	lu2	Logical unit number of the model run log file.
	lu3	Logical unit number of the direct access file.
	dt	Model time step.
	iend	Nesting file flag. Set to one after last nesting file is read.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.16 Subroutine PARAMS

This subroutine reads the MAPS.DAT file and sets various constants and parameters. It is called by the MAIN program.

Calling sequences: subroutine const(lpar, lu2)

Data Declarations: logical lpar, lu2

Arguments:		
	lpar	Logical unit number of the model run log file.
	lu2	Logical unit number of the direct access file.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.17 Subroutine SEMIMP

SEMIMP sets up the Helmholtz equation to solve for TP and also solves for UP, VP, and PSP. It is called by the MAIN program.

Calling sequences: subroutine semimp(dt)

Data Declarations: real dt

Arguments:		
	dt	Model time step.

Common Blocks: DOUBLE, THREED, TWOD, ONEDR, ONEDC

5.1.3.18 Subroutine TRIDAG

TRIDAG is a tridiagonal matrix solver. It is called by FIZZIX to solve the vertical diffusion equation.

Calling sequences: subroutine tridag(a, n, ndim)

Data Declarations: real a
integer n, ndim

Arguments:		
	a	*a(ndim, 4)* is the array to be inverted.
	n	The number of elements stored in first dimension of *a*.
	ndims	The dimension of the first array element of *a*.

5.1.3.19 Function FF

FF is used in the calculation of the Monin-Obukhov length L. It is called by FIZZIX.

Calling sequences: function ff(zl, z0l)

Data Declarations: real zl, z0l

Arguments: zl — Lowest layer depth divided by Monin-Obukhov length.

z0l — Roughness length (z0) divided by Monin-Obukhov length.

5.1.3.20 Function GG

GG is used in the calculation of the Monin-Obukhov length L. It is called by FIZZIX.

Calling sequences: function gg (zl, zhl)

Data Declarations: real zl, zhl

Arguments: zl — Lowest layer depth divided by Monin-Obukhov length.

zhl — zh divided by Monin-Obukhov length.

5.1.3.21 Function PSIM

This is a profile stability function for momentum. It is called by FIZZIX for Monin-Obukhov surface layer calculation.

Calling sequences: function psim (zl)

Data Declarations: real zl

Arguments: zl — Lowest layer depth divided by Monin-Obukhov length.

5.1.3.22 Function PSIH

This is a profile stability function for heat. It is called by FIZZIX for Monin-Obukhov surface layer calculation.

Calling sequences: function psih (zl)

Data Declarations: real zl

Arguments: zl — Lowest layer depth divided by Monin-Obukhov length.

5.1.3.23 Function PHIMI

PHIMI calculates ϕ_M for Monin-Obukhov calculation. It is called by FIZZIX.

Calling sequences: function phimi (zl)

Data Declarations: real zl

Arguments: zl — Lowest layer depth divided by Monin-Obukhov length.

Algorithm:
Refer to Monin-Obukhov Surface Layer, Section 5.1.2.7.

5.1.3.24 Function PHIHI

PHIMI calculates ϕ_H for Monin-Obukhov calculation. It is called by FIZZIX.

Calling sequences: function phihi (zl)

Data Declarations: real zl

Arguments: zl — Lowest layer depth divided by Monin-Obukhov length.

Algorithm:
Refer to Monin-Obukhov Surface Layer, Section 5.1.2.7.

5.1.3.25 Function ESAT

ESAT calculates the saturated vapor pressure for a given temperature. It is called by FIZZIX.

Calling sequences: function esat (t)

Data Declarations: real t

Arguments: t — Temperature for which the saturated vapor pressure is required.

The organization of subroutines and functions within the MAPS model is illustrated in Figure 5.1.3-1.

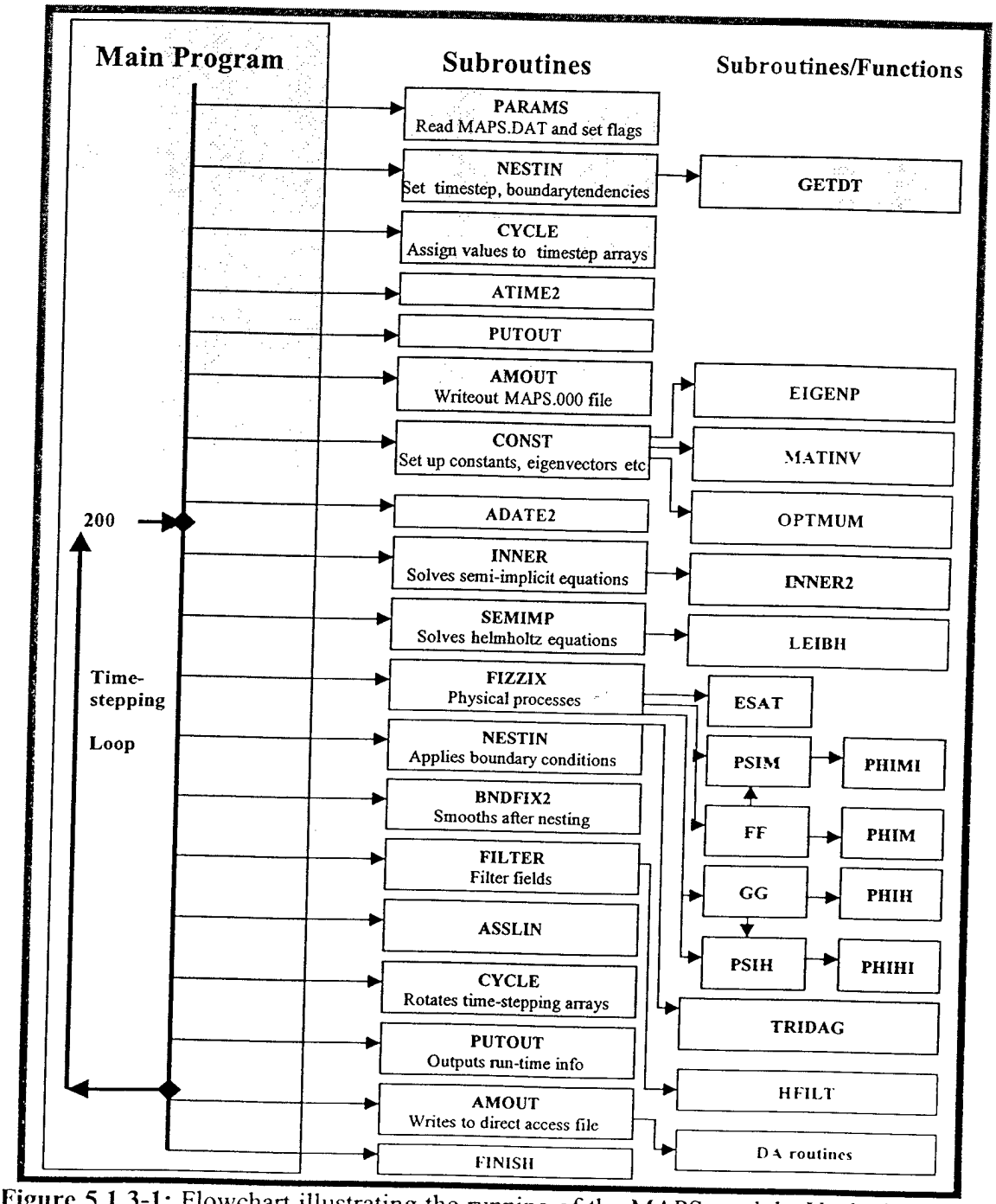

Figure 5.1.3-1: Flowchart illustrating the running of the MAPS model. Unshaded subprograms indicate those stored in separately compiled libraries. Note also that multiple calls to subprograms within subroutines are not indicated to simplify the flow diagram.

5.1.4 The MAPS System

In addition to the MAPS model, several other modules and programs are necessary to complete the modeling system. These include programs for setting up model grids over specified geographic regions, extraction and conversion programs to provide atmospheric initial and boundary conditions in suitable format for MAPS and display programs for viewing model output. These additional programs are summarized in Figure 5.1.4-1 while the key additional elements are described in the following sections.

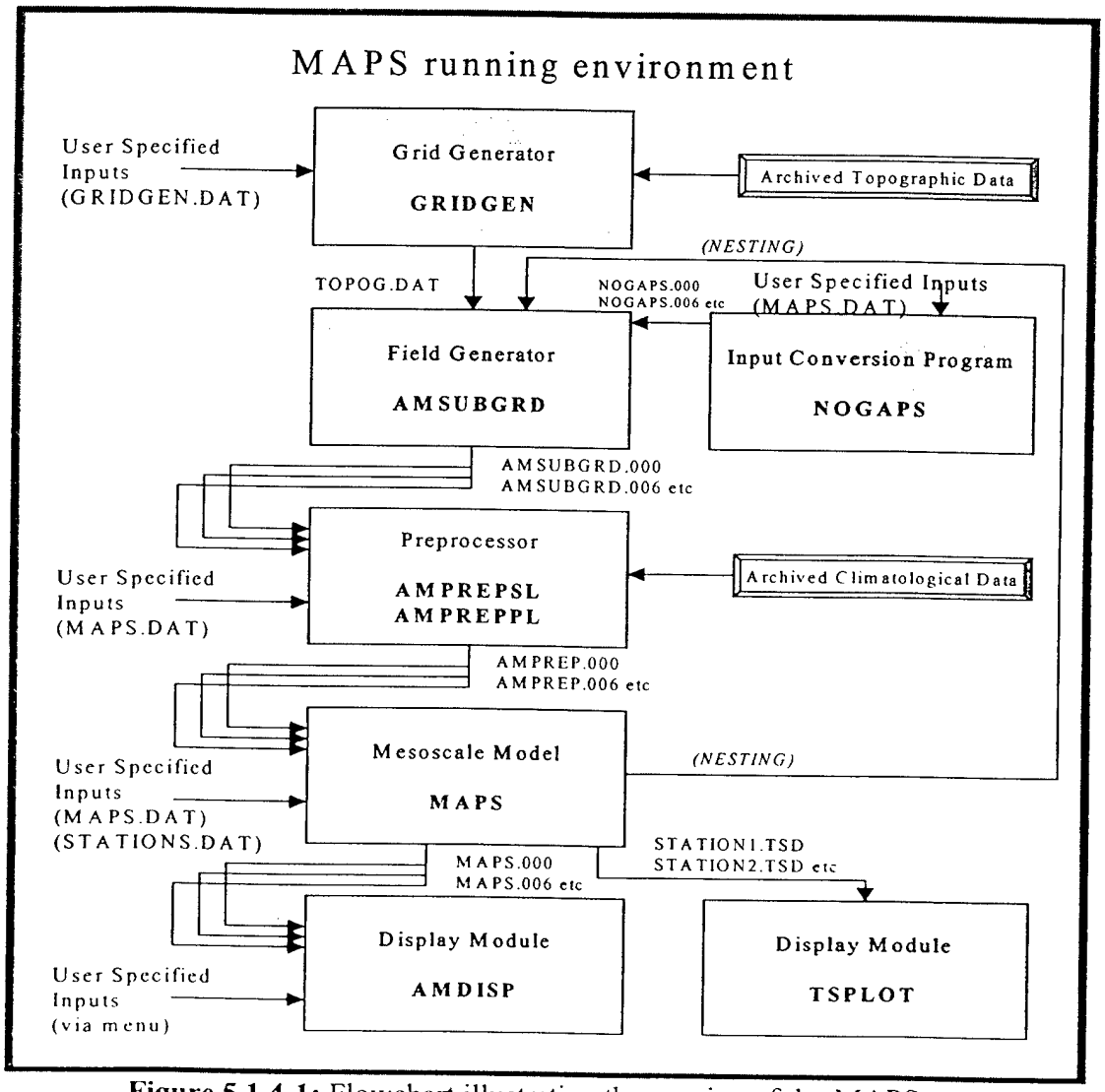

Figure 5.1.4-1: Flowchart illustrating the running of the MAPS system.

5.1.4.1 The Grid Generator

The grid generator sets up a model grid over a designated region and extracts the topographic and bathymetric data and writes a TOPOG.DAT file.

5.1.4.2 Input Conversion Program

The input conversion program extracts the various required fields from the NOGAPS model and outputs all necessary variables over a three-dimensional region at each time in direct access format. The output files are written for each time covering the duration of the required simulation. The naming convention for the files is NOGAPS.000 for the zeroth hour (analysis) data file, NOGAPS.006 for the 6th hour data file and so on.

5.1.4.3 The Field Generator

The field generator reads the TOPOG.DAT file and inputs the atmospheric data files sequentially (either derived from the NOGAPS model or a coarse resolution simulation of MAPS) and horizontally interpolates the fields to the region defined by the TOPOG.DAT parameters. The output files are written in direct access format to AMSUBGRD.nnn, where "nnn" is a three-digit integer denoting the number of hours from initial time

5.1.4.4 The Preprocessor

The preprocessor reads the MAPS.DAT parameter file to obtain the vertical levels required for the MAPS simulation and interpolates the AMSUBGRD files to the appropriate vertical levels. There are two versions of the preprocessor, AMPREPPL and AMPREPSL. The first of these assumes input on pressure levels and interpolates the AMSUBGRD.nnn files from pressure levels to sigma levels, the second of these assumes input on sigma levels if the input atmospheric data files originated from a coarse resolution MAPS simulation. The preprocessor outputs a series of AMPREP.nnn files at the appropriate time interval for the duration of the run.

5.1.4.5 Display Modules

There are options for displaying MAPS model output. The first of these is AMDISP that plots the simulated variables at designated levels and times during the model simulation and the second TSPLOT that displays the time varying station output.

5.2 CSC GCOM2D

CSC GCOM2D is a depth-integrated, barotropic hydrodynamic model used for modeling sea surface height and mean current structure in coastal regions due to tides and atmospheric forcing. It is economic to run in terms of computational overhead and data initialization requirements.

5.2.1 Constraints and Limitations

GCOM2D/3D has been designed such that total grid arrays in the horizontal direction are no larger than 40000 points (for example, 200x200).

5.2.2 Logic and Basic Equations

The equations solved are the shallow water equations and are presented below;

$$\frac{\partial U}{\partial t} = fV - mg\frac{\partial \zeta}{\partial x} - \frac{m}{\rho_\omega}\frac{\partial P}{\partial x} - m\left(U\frac{\partial U}{\partial x} + V\frac{\partial U}{\partial y}\right) + \frac{1}{\rho_\omega H}\left(\tau_{sx} - \tau_{bx} - \frac{\partial S_{xx}}{\partial x} - \frac{\partial S_{xy}}{\partial y}\right) - \nu\nabla^2 U, \tag{5.2.1}$$

$$\frac{\partial V}{\partial t} = -fU - mg\frac{\partial \zeta}{\partial y} - \frac{m}{\rho_\omega}\frac{\partial P}{\partial y} - m\left(U\frac{\partial V}{\partial x} + V\frac{\partial V}{\partial y}\right) + \frac{1}{\rho_\omega H}\left(\tau_{sy} - \tau_{by} - \frac{\partial S_{yx}}{\partial x} - \frac{\partial S_{yy}}{\partial y}\right) - \nu\nabla^2 V, \tag{5.2.2}$$

$$\frac{\partial \zeta}{\partial t} = -m^2\left[\frac{\partial}{\partial x}\left(\frac{UH}{m}\right) + \frac{\partial}{\partial y}\left(\frac{VH}{m}\right)\right], \tag{5.2.3}$$

where U and V are the depth averaged currents in the x and y directions respectively, H is the total depth, ζ is the surface elevation, f is the Coriolis parameter, g is the acceleration due to gravity, P is the atmospheric surface pressure, ρ_w is the water density, ν is the coefficient of lateral eddy diffusion and has a value of 0.2, τ_{sx} and τ_{by}, the bottom frictional stress in the x and y directions, respectively. S_{xx}, S_{yy}, S_{xy} and S_{yx} represent the wave radiation stresses.

The equations have been formulated on a Cartesian grid due to the relative ease of coding. However, the formulation is generalized to incorporate different map projections through the appropriate specification of the map factor, m, a scaling factor dependent on the chosen map projection of the model grid. In accordance with this generality, the Coriolis parameter varies with latitude.

The surface wind stress components are computed using the quadratic relationship:

$$\tau_{sx} = C_D \rho_a |\mathbf{u_a}| u_a, \qquad \tau_{sy} = C_D \rho_a |\mathbf{u_a}| v_a, \tag{5.2.4}$$

where $|\mathbf{u_a}| = \left(u_a^2 + v_a^2\right)^{1/2}$ u_a and v_a are the horizontal components of wind velocity at anemometer height, ρ_a is the density of air, and C_D is the drag coefficient based on Smith and Banke (1975) and expressed as follows:

$$\begin{aligned} C_D &= \left[0.63 + 0.066|\mathbf{u_a}|\right] \times 10^{-3}, && |\mathbf{u_a}| < 25\,\mathrm{m\,s^{-1}}; \\ C_D &= \left[2.28 + 0.033\left(|\mathbf{u_a}| - 25\right)\right] \times 10^{-3}, && |\mathbf{u_a}| \geq 25\,\mathrm{m\,s^{-1}}. \end{aligned} \tag{5.2.5}$$

The bottom stress is represented by a Manning's n depth-dependent friction relation following Signell and Butman (1992):

$$\tau_{bx} = \rho_w \frac{gn^2}{(H+\zeta)^{1/3}} \left(U^2 + V^2\right)^{1/2} U, \qquad \tau_{by} = \rho_w \frac{gn^2}{(H+\zeta)^{1/3}} \left(U^2 + V^2\right)^{1/2} \tag{5.2.6}$$

where n has the value 0.03. This formulation ensures that the drag coefficient increases with decreasing water depth and is applied to water depths greater than 1 m. In extremely shallow water and over land points that become inundated, drag coefficients can be specified at each grid-point according to the terrain type.

5.2.2.1 Solution Procedure

Equations (5.2.1) to (5.2.3) are solved using a split-explicit finite-difference scheme on an Arakawa-C grid (Messinger and Arakawa, 1976) as shown in Figure 5.2.2-1 and described in Hubbert et al. (1990). The continuity equation and the gravity wave and Coriolis terms in the momentum equations are solved on the shortest time step, known as the *adjustment* step, using the forward-backward method. The non-linear advective terms are solved on an intermediate *advective* time step using the two-time-level method of Miller and Pearce (1974). Finally, on the longest time step, the so-called *physics* step, the surface wind stress, bottom friction stress and atmospheric pressure terms are solved using a backward-implicit method. This approach is extremely efficient in oceanographic models with free surfaces because of the large disparity between advective speeds and gravity-wave phase speeds in deep water.

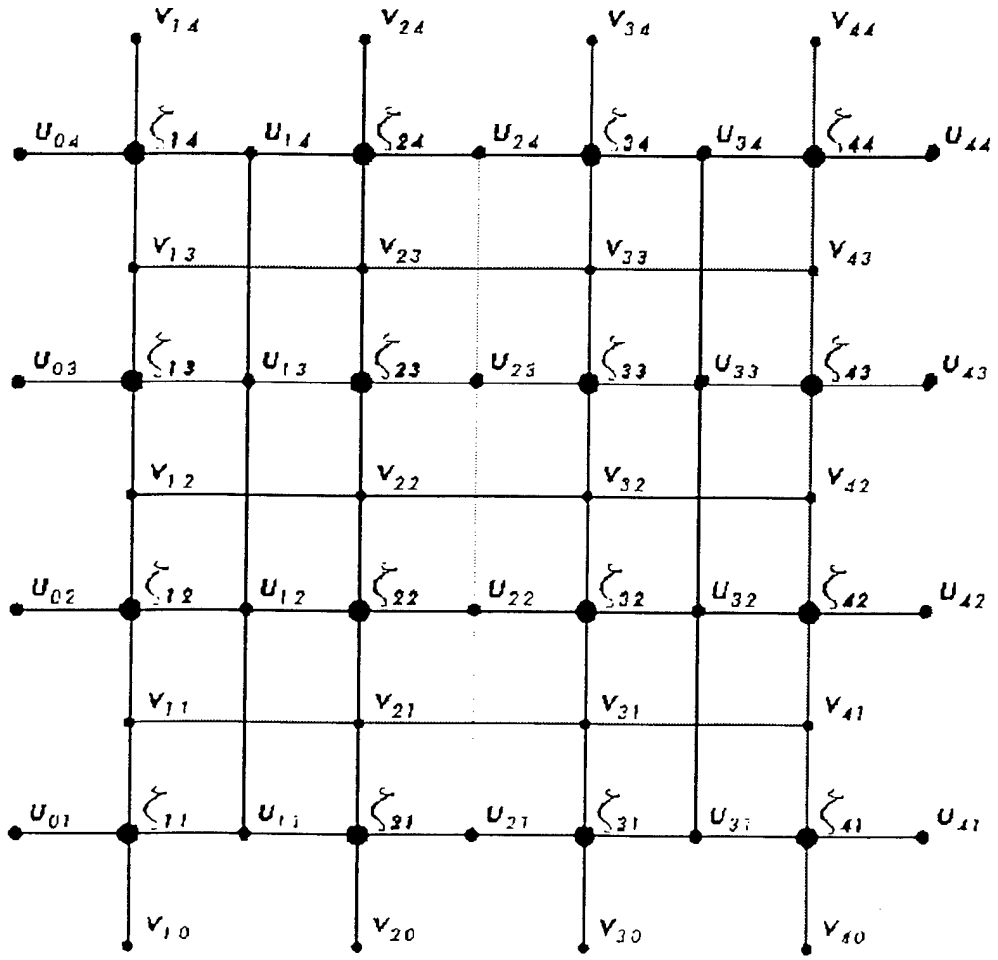

Figure 5.2.2-1: The horizontal grid structure used in GCOM2D.

(a) Adjustment step:
By defining

$$m_{ij}^{u} = (m_{ij} + m_{i+1j})/2$$
$$m_{ij}^{v} = (m_{ij} + m_{ij+1})/2$$
$$H_{ij}^{u} = (H_{ij} + H_{i+1j})/2$$
$$H_{ij}^{u} = (H_{ij} + H_{i+1j})/2$$

and noting that the grid spacing at the standard latitude of the map projection is represented by $\Delta s = \Delta x = \Delta y$, the adjustment step is accomplished by the following:

$$\zeta_{ij}^{t+\Delta t} = \zeta_{ij}^{t} - m_{ij}^{2}\frac{\Delta t}{\Delta s}\left\{\frac{(UH^{u})_{ij}^{t}}{m_{ij}^{u}} - \frac{(UH^{u})_{i-1j}^{t}}{m_{i-1j}^{u}} + \frac{(VH^{v})_{ij}^{t}}{m_{ij}^{v}} - \frac{(VH^{u})_{ij-1}^{t}}{m_{ij-1}^{v}}\right\} \quad (5.2.7)$$

The velocity components are then computed using the latest values of the surface elevation in the implicit relationship;

$$U_{ij}^{*} = U_{ij}^{t} + \Delta t\left\{ fV_{ij}^{*} - m_{ij}^{u} g\left(\frac{\partial \zeta}{\partial x}\right)_{ij}^{t+\Delta t} \right\} \tag{5.2.8}$$

$$V_{ij}^{*} = V_{ij}^{t} - \Delta t\left\{ fV_{ij}^{*} + m_{ij}^{v} g\left(\frac{\partial \zeta}{\partial y}\right)_{ij}^{t+\Delta t} \right\} \tag{5.2.9}$$

These equations may be rewritten directly in the explicit form to obtain the intermediate solutions U^{*} and V^{*}:

$$U_{ij}^{*} = \frac{1}{(1+f^{2}\Delta t^{2})}\left[U_{ij}^{t} - m_{ij}^{u} g\Delta t\left(\frac{\partial \zeta}{\partial x}\right)_{ij}^{t+\Delta t} + f\Delta t\left\{ \overline{V}_{ij}^{t} - m_{ij}^{u} g\Delta t\left(\frac{\partial \zeta}{\partial y}\right)_{ij}^{t+\Delta t} \right\} \right] \tag{5.2.10}$$

$$V_{ij}^{*} = \frac{1}{(1+f^{2}\Delta t^{2})}\left[V_{ij}^{t} - m_{ij}^{v} g\Delta t\left(\frac{\partial \zeta}{\partial y}\right)_{ij}^{t+\Delta t} - f\Delta t\left\{ \overline{U}_{ij}^{t} - m_{ij}^{v} g\Delta t\left(\frac{\partial \zeta}{\partial x}\right)_{ij}^{t+\Delta t} \right\} \right] \tag{5.2.11}$$

Where

$$\overline{U}_{ij}^{t} = (U_{ij}^{t} + U_{ij+1}^{t} + U_{i-1j}^{t} + U_{i-1j+1}^{t})/4$$

$$\overline{V}_{ij}^{t} = (V_{ij}^{t} + V_{i+1j}^{t} + V_{ij-1}^{t} + V_{i+1j-1}^{t})/4$$

Equations (5.2.7), (5.2.10) and (5.2.11) are solved N_a times so that the adjustment step advances the intermediate solution to the values U^{*} and V^{*} at time $t + \Delta t_a$, where $\Delta t_a = N_a \Delta t$.

(b) Advective step:

The numerical scheme chosen for the advective step is the two-time-level method of Miller and Pearce (1974). This scheme alternates the Euler and Euler-backward schemes at odd and even advective time-steps and has the major advantage of an amplification factor that is almost exactly unity for the Courant numbers that are found in ocean models (see Hubbert et al. 1991).

On the odd advective time-steps, the updated intermediate solutions U^{**} and V^{**} are

$$U_{ij}^{**} = (U_{ij}^{*} + \Delta t_a F(U_{ij}^{*})_a \tag{5.2.12}$$

$$V_{ij}^{**} = (V_{ij}^{*} + \Delta t_a G(V_{ij}^{*})_a \tag{5.2.13}$$

where the advective operators F and G are defined by

$$F(U_{ij}^{*}) = -\frac{m_{ij}^{u}}{4\Delta s}\{(U_{i+1j}^{*}+U_{ij}^{*})(U_{i+1j}^{*}-U_{ij}^{*})+(U_{ij}^{*}+U_{i-1j}^{*})(U_{ij}^{*}-U_{i-1j}^{*})+$$
$$(V_{i+1j}^{*}+V_{ij}^{*})(U_{ij+1}^{*}+U_{ij}^{*})+(V_{ij-1}^{*}+V_{i+1j-1}^{*})(U_{ij}^{*}+U_{ij-1}^{*})\}$$

$$G(V_{ij}^{*}) = -\frac{m_{ij}^{u}}{4\Delta s}\{(U_{ij+1}^{*}+U_{ij}^{*})(V_{i+1j}^{*}-V_{ij}^{*})+(U_{i-1j}^{*}+U_{i-1j+1}^{*})(V_{ij}^{*}-V_{i-1j}^{*})+$$
$$(V_{ij+1}^{*}+V_{ij}^{*})(V_{ij+1}^{*}-V_{ij}^{*})+(V_{ij}^{*}+V_{ij-1}^{*})(V_{ij}^{*}+V_{ij-1}^{*})\}$$

On the even advective time-steps, the two-step Euler-backward scheme is applied instead of equations (5.2.12) and (5.2.13). On the second iteration, the values of U and V obtained after the first iteration are substituted into the advective operators F and G to obtain the final values of U^{**} and V^{**}.

(c) Physics step:

The adjustment and advective integration cycle is carried out N_p times so that the interim solution is U^{**} and V^{**} at time $t+\Delta t_p$, where $\Delta t_p = N_p \Delta t_a = N_p N_a \Delta t$. At this time, the solution cycle is completed with the inclusion of the physics terms using a numerical technique similar to that described for the adjustment step. The final value of U is therefore

$$U_{ij}^{t+\Delta t_p} = U_{ij}^{**} + \frac{\Delta t_p}{\rho_w}\left[\frac{1}{H_{ij}^{u}}(\tau_{sx}-\tau_{bx}) - \frac{m_{ij}^{u}}{\Delta s}(P_{i+1j}^{a}-P_{ij}^{a})\right] \qquad (5.2.14)$$

$$= U_{ij}^{**} + \frac{\Delta t_p}{\rho_w}\left[\frac{\tau_{sx}}{H_{ij}^{u}} - \frac{\rho_w}{H_{ij}^{u}} K U_{ij}^{t+\Delta t_p}(U_{ij}^{**2}+\overline{V}_{ij}^{**2})^{2} - \frac{m_{ij}^{u}}{\Delta s}(P_{i+1j}^{a}-P_{ij}^{a})\right]$$

Rewriting in explicit form gives

$$U_{ij}^{t+\Delta t_p} = U_{ij}^{**} + \frac{\Delta t_p}{\rho_w}\left[\frac{\tau_{sx}}{H_{ij}^{u}} - \frac{m_{ij}^{u}}{\Delta s}(P_{i+1j}^{a}-P_{ij}^{a})\right] \Bigg/ \left[1+\frac{\Delta t_p}{H_{ij}^{u}} K (U^{**2}+\overline{V}_{ij}^{**2})^{1/2}\right]$$

Similarly

$$V_{ij}^{t+\Delta t_p} = V_{ij}^{**} + \frac{\Delta t_p}{\rho_w}\left[\frac{\tau_{sy}}{H_{ij}^{v}} - \frac{m_{ij}^{v}}{\Delta s}(P_{ij+1}^{a}-P_{ij}^{a})\right] \Bigg/ \left[1+\frac{\Delta t_p}{H_{ij}^{v}} K (\overline{U}^{**2}+V_{ij}^{**2})^{1/2}\right]$$

The numerical scheme therefore is split-explicit and consists of three distinct time steps Δt, Δt_a and Δt_p where $\Delta t \le \Delta t_a \le \Delta t$.

5.2.2.2 Boundary and Initial Conditions

Boundary conditions can be applied in a range of ways depending on the type of process being modeled. Meteorological forcing is applied via the wind stress and surface pressure gradient terms in equations (5.2.1) to (5.2.3) at all submerged model grid-points in the computational domain.

Tidal and meteorological forcing at lateral boundaries is achieved by specifying the incremental displacement of the water surface due to changes in tidal height and atmospheric barometric displacement. The lateral boundary conditions are applied using a "one-way nesting" technique. The boundary conditions are applied to the appropriate model variable ϕ (representing height or velocity) with decreasing intensity from the boundary to some specified number of model grid-points (typically around 12) into the domain according to the following equation;

$$\phi = (1-\alpha)\phi_p + \alpha\phi_b \qquad (5.2.15)$$

where ϕ_p is the model predicted value and ϕ_b is the prescribed boundary value and α is varied according to a cosine function such that

$$\alpha = 0.5(\cos \pi(1\text{-}n/n_{max}) + 1), \quad n=1, n_{max} \qquad (5.2.16)$$

At coastal boundaries and along riverbanks, the wetting and drying of grid cells is accomplished via the inundation algorithm described in the next section. On outflow, a radiation boundary condition, as described in Miller and Thorpe (1981) is applied to the velocity field to prevent the build up of wave energy within the numerical domain, while on inflow boundaries, a zero-gradient condition is applied.

Under any prescribed boundary forcing, GCOM2D is initialized by setting velocities to zero and interpolates the global Finite Element Solutions versions 95.1/2.1 (FES 95.1/2.1) (Shum et al, 1997) elevation field to the model grid. It is customary to allocate an initial spin-up period to allow the effects of the boundary forcing and tidal conditions to propagate throughout the computational domain. The spin-up time will vary according to domain size but would typically be 12 hours without winds and 24 hours with winds.

5.2.2.3 Inundation Algorithm

The adjustment of coastlines to account for flooding and draining takes place after equations (5.2.1)-(5.2.3) have been solved as described above. The coastal boundary is configured to pass through the velocity grid-points on the staggered grid in a stepwise manner such that it passes through the U points in the y-direction and through V points in the x-direction. The velocities on the boundaries are assumed to be zero.

The first step is to calculate the distance in the x- and y-directions that fluid could travel in a time step at each ζ grid-point that is adjacent to the coast. The depth-averaged current velocity used in this calculation is taken at the first grid-point on the seaward side of the ζ grid-point. The travel distance is:

$$\Delta X_{i,j}^{n} = \begin{cases} \Delta X_{i,j}^{n} + \Delta t U_{i-1,j}^{n}, U > 0 \\ \Delta X_{i,j}^{n-1} + \Delta t U_{i,j}^{n}, U < 0 \end{cases} \tag{5.2.17}$$

$$\Delta Y_{i,j}^{n} = \begin{cases} \Delta Y_{i,j}^{n-1} + \Delta t V_{i,j-1}^{n}, V > 0 \\ \Delta Y_{i,j}^{n-1} + \Delta t V_{i,j}^{n}, V < 0 \end{cases} \tag{5.2.18}$$

where $\Delta X_{i,j}^{n-1}$ and $\Delta Y_{i,j}^{n-1}$ are the distances in the x and y directions that the water traveled in the previous time step. By factoring in the travel time of the fluid, inland grid cells are prevented from automatically becoming inundated at the first instant the water level at the coast exceeds the height of the adjacent dry points. In equations (5.2.17) and (5.2.18), the first option applies to flooding in the positive x or y direction (or draining in the negative x and y direction) while the second option applies to flooding in the negative x and y direction (or draining in the positive x and y direction).

The testing for coastline movement proceeds in the x- and y-directions separately. If the height of the water at the first z-point seaward of the coastline exceeds the topographic height at the first z-point landward of it, and the accumulated distance traveled in the given direction exceeds the grid separation, then a new sea-point is added to the computational domain. The velocity at the newly acquired velocity point is extrapolated from adjacent points and the continuity equation is solved to obtain the water depth at the new height point. Finally, $\Delta X_{i,j}^{n}$ and $\Delta Y_{i,j}^{n}$ are set to zero.

The procedure for draining is similar. If the height of the fluid at the height point adjacent the boundary drops below some arbitrary positive height, ε and the accumulated distance traveled by the fluid, exceeds the grid length, then draining is assumed to have occurred. The height grid-point is reclassified as dry (i.e. $\zeta = 0$) and the boundary relocated to the adjacent wet velocity grid-point. Examples of the performance of the inundation algorithm can be found in Hubbert and McInnes (1999a, b)

5.2.2.4 Tidal Data Assimilation

In order to improve the simulation of tidal forced dynamics a facility is included to "nudge" the model solution with tidal height predictions at tidal stations within the model domain. Global tidal station constituent data is stored in ASCII format in the file TCANALS.DAT. New stations can easily be added by adding to this file in the appropriate format.

During a model run the TCANALS.DAT file is scanned by subroutine TIDEOBS during its first call and the tidal constituent parameters for stations within the model domain are stored. At each model time step the tidal height is predicted at each station and used to "nudge" the model solution.

The nudging method is based on deriving a new solution at grid points near each tidal station from a weighted combination of the model solution and the station sea level prediction. The weighting function is calculated from the product of the parameter WGTFAC (default value = 0.50) and the half cosine function (range = 0 to 1) used for nesting (equation 5.2.16). The weighting of the station sea level prediction goes to zero at a defined distance from each station, determined by the parameter ZRADIUS (default value = 40 kms). The values of WGTFAC and ZRADIUS can be changed in the parameter file ASSIM.DAT that resides in the WORK directory. Changes to these parameters are not advised however unless there is a specific reason.

5.2.3 Data Elements

5.2.3.1 Commonly Used Parameters

File: 2dparams.for

Variable	Type	Description
nk	Integer	Maximum vertical dimension.
ni	Integer	Maximum horizontal dimension (x-direction).
nj	Integer	Maximum horizontal dimension (y-direction).
nip1	Integer	NI+1.
njp1	Integer	NJ+1.

5.2.3.2 Subroutine ADVECT

ADVECT calculates the advective terms of the momentum and continuity equations. It is called by the MAIN program.

Calling Sequence: subroutine advect(nx, ny,dt)

Data Declaration: real dt
integer nx, ny

Arguments:

nx		Array dimensions in x-direction.
ny		Array dimensions in y-direction.
dt		Time step.

Common Blocks: 2DCOMMON

5.2.3.3 Subroutine ASTROZ

This routine calculates the frequency (in cycles/hour), the astronomical arguments (in cycles), and the amplitude for the given tidal constituents. It is called by TIDEOBS.

Calling Sequence: subroutine astroz (lstc, con, freq, indx, utc, vtc, ftc, ntcmax, ncons, nobs, hour, xlat)

Data Declaration: real con, freq, utc, vtc, ftc, hour, xlat
integer lstc, indx, ntcmax, ncons, nobs

Arguments:		
	lstc	Logical unit which reads the standard tidal constituent data from STCONTHP.DAT.
	con	Tidal constituents array; *con (ntcmax, nobs).*
	freq	Frequencies of tidal constituents calculated in ASTROZ.
	indx	Stored look-up table indices from frequency calculation.
	utc	Nodal correction phase calculated in ASTROZ.
	vtc	Astronomical argument in cycles calculated in ASTROZ.
	ftc	Amplitude of tidal constituents calculated in ASTROZ.
	ntcmax	Maximum number of tidal constituents.
	ncons	Number of tidal constituents available at each tidal station.
	nobs	Number of station observations available.
	hour	Double precision hour relative to time origin (01-01-1900).
	xlat	Station latitude.

Common Blocks: 2DCOMMON

5.2.3.4 Subroutine ATMOS

ATMOS reads the meteorological input file and calculates the wind stress and atmospheric pressure. It is called by the MAIN program.

Calling Sequence: subroutine atmos(lu, wdfile, istart, wdmax, rhoa, punits, wtzone, ihour, iminut, iday, imonth, iyear, nx, ny,dt)

Data Declaration: real wdfile, wdmax, rhoa, punits, wtzone, dt
integer nx, ny,
integer istart, ihour, iminut, iday, imonth, iyear
logical lu

Arguments:		
	lu	Logical unit number of the model run log file.
	wdfile	Name of meteorological input file.
	istart	No. of hours of model spin-up time.
	wdmax	Wind speed cut-off in wind stress formulation.
	rhoa	Density of air.
	punits	Multiplier to convert pressure from pascals into mks units.
	wtzone	Wind time zone, in real hours relative to GMT.
	ihour	2 digit integer for current hour.
	iminut	2 digit integer for current minute.

iday	2 digit integer for current day.
imonth	2 digit integer for current month.
iyear	2 digit integer for current year.
nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.
dt	Time step.

Common Blocks: 2DCOMMON

5.2.3.5 Subroutine CYCLE

Subroutine CYCLE rotates time-stepping arrays at the end of each time step. It is called by MAIN program.

Calling Sequence: subroutine cycle(nx, ny)

Data Declaration: integer nx, ny

Arguments:

nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.

Common Blocks: 2DCOMMON

5.2.3.6 Subroutine ENERGY

ENERGY integrates the total energy of the system as a diagnostic performance indicator. It is called by the MAIN program.

Calling Sequence: subroutine energy(nx, ny,egp)

Data Declaration: real egp
integer nx, ny

Arguments:

nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.
egp	The total energy of the system.

Common Blocks: 2DCOMMON

5.2.3.7 Subroutine FILTZ

FILTZ applies a five-point filter to the sea level array to remove two-grid-length noise. It is called by the MAIN program.

Calling Sequence: subroutine filtz(nx, ny, fltfac, spv)

Data Declaration: real fltfac, spv
integer nx, ny

Arguments:		
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	fltfac	Inactive argument.
	spv	Special value. No filtering is applied when height = *spv*.

Common Blocks: 2DCOMMON

Algorithm:

See section 5.7- Filtering Routines

5.2.3.8 Subroutine FILTUV

FILTUV applies a five-point filter to the velocity arrays to remove two-grid-length noise. It is called by the MAIN program.

Calling Sequence: subroutine filtuv(nx, ny, fltfac, spv)

Data Declaration: real fltfac, spv
integer nx, ny

Arguments:		
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	fltfac	Inactive argument.
	spv	Special value. No filtering is applied when height = *spv*.

Common Blocks: 2DCOMMON

5.2.3.9 Subroutine GRDSET

GRDSET sets up the staggered bathymetry, coastlines and vertical levels information (GCOM3D only). It is called by the MAIN program.

Calling Sequence: subroutine grdset(nx, ny,dxymin, dxymax, plat0, plong0, plat1, plong1)

Data Declaration: real dxymin, dxymax, plat0, plong0, plat1, plong1
integer nx, ny

Arguments:		
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	dxymin	Value of minimum grid spacing on grid for log file.
	dxymax	Value of maximum grid spacing on grid for log file.
	plat0	Latitude of minimum grid spacing (reporting variable).
	plong0	Longitude of minimum grid spacing (reporting variable).
	plat1	Latitude of maximum grid spacing (reporting variable).
	plong1	Longitude of maximum grid spacing (reporting variable).

Common Blocks: 2DCOMMON

5.2.3.10 Subroutine MASSBAL

The subroutine monitors the mass balance throughout the model integration as a diagnostic performance indicator.

Calling Sequence: subroutine massbal(nx, ny,dt, cum_sqkm, iplot)

Data Declaration: real dt, cum_sqkm
integer nx, ny,iplot

Arguments:		
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	dt	Model time step.
	cum_sqkm	Redundant variable.
	iplot	Real-time output flag (0=text or 1= graphics to screen).

Common Blocks: 2DCOMMON

The total volume of the computational domain at any time is given by:

$$V_{o:} = \sum_{j=1}^{mv} \sum_{i=1}^{mv} dx\, dy (H + \zeta)_{ij} \qquad (5.2.3.10a)$$

The mass flux through the lateral boundary is calculated from equation (5.2.3.10a) over a time interval *dt*. The total mass added to the system is monitored by comparing equations (5.2.3.10a) and (5.2.3.10b).

$$V_{ol} = \sum_{i=o}^{ny} U_{0,j} \times dy \times (H+\zeta)_{0,j} \times dt + \sum_{j=o}^{ny} U_{nx,,j} \times dy \times (H+\zeta)_{nx,j} \times dt + \sum_{i=o}^{nx} V_{i,0} \times dy \times (H+\zeta)_{i,0} \times dt + \sum_{i=o}^{nx} V_{i,ny} \times dy \times (H+\zeta)_{i,ny} \times dt \quad (5.2.3.10b)$$

The change in volume dV between any two time steps n and $n+1$ is given by

$$dV_{ol} = V_{ol}^{n+1} - V_{ol}^{n} \quad (5.2.3.10c)$$

5.2.3.11 Subroutine PARAMS

PARAMS reads the parameter file GCOM.DAT and the time series output stations file STATIONS.DAT. It then determines which tidal files are present (if required) and reads the initial wind data (if required).

Calling Sequence: subroutine params(lu, lwd, llog, bfile, rsfile, wdfile. tcfile, wdmax, rhoa, punits, iwind, itide, iassim, nstpts, istart. inund, iplot, idrift, ibc. imass, iflux, nadv, nphys, ntcmax, ntcs, ntca. nstmax, sponge, datum, fltfac, dtfac, wdt, wtzone, depmin, depmax, hc1. hc2, hc3, hc4, hc5, ihour, iminut, iday, imonth, iyear, hrout, dtout, nhours, dhour, dminut, dday. dmonth, dyear, dlat, dlong, sptime, stname, stnlat, stnlng, stdep, thetap, ip. jp, stni, stnj, nstns, region, jproj, proj, height, nx. ny)

Data Declaration: real dt, wdmax, rhoa, punits, sponge, datum, fltfac. dtfac, wdt, wtzone, depmin, depmax, hc1, hc2. hc3, hc4, hc5, hrout, dtout, dlat, dlong, sptime, stnlat, stnlng, stdep, thetap. stni, stnj, , proj, height
integer nx, ny,iwind, itide, iassim, nstpts, istart. inund, iplot, idrift,
ibc, imass,iflux, nadv, nphys, ntcmax, ntcs, ntca. nstmax,
ihour, iminut, iday. imonth iyear, nhours, ip. jp, nstns, jproj, dhour. dminut, dday. dmonth, dyear,
logical lu, lwd, llog
char bfile, rsfile, wdfile, tcfile, stname, region

Arguments:

lu	Logical unit of input parameter file.
lwd	Logical unit of wind file (ATMOS.DAT).
llog	Logical unit of output log file
bfile	Topography file name ('TOPOG.DAT').

rsfile	Restart file name ('GCOM.RST').
wdfile	Atmospheric input file name ('ATMOS.DAT').
tcfile	Tidal constituent file name ('M2.DAT, S2.DAT', etc.).
wdmax	Wind speed cut-off in wind stress formulation.
rhoa	Density of air.
punits	Multiplier to convert pressure from pascals into mks units.
iwind	Wind forcing flag (0=no wind forcing, 1= wind forcing).
itide	Tide forcing flag (0=not tidal forcing, 1= tidal forcing, 2= assimilate data).
iassim	Tidal assimilation flag (1= assimilated tidal information).
nstpts	Number of points for nesting in (set internally).
istart	Number of points for nesting in (set internally).
inund	Inundation flag (1=call inundation algorithm).
iplot	Screen or text flag.
idrift	Inactive flag.
ibc	Boundary condition flag (set to 5).
imass	Flag for calculating mass balance (set to 0).
iflux	Inactive flag.
nadv	Number of gravity wave steps before advection step.
nphys	Number of gravity wave steps before physics step.
ntcmax	Maximum number of tidal constituents.
ntcs	Number of tidal constituents available at each tidal station.
ntca	Inactive.
nstmax	Maximum number of stations.
sponge	Inactive.
datum	Inactive.
fltfac	Inactive.
dtfac	CFL reduction factor (internally set 0.95).
wdt	time stepping.
wtzone	Wind time zone, in real hours relative to GMT.
depmin	Minimum allowable depth in model (set in PARAM).
depmax	Maximum allowable depth in model (set in PARAM).
hc1, hc2, hc3, hc4, hc5	Heights that define color bands for screen graphics.
ihour, iminut, iday imonth, iyear	Initialization time of model.
hrout	Initial output time (set at 0).
dtout	Time-step of GCOM.OUT file.
nhours	total number of hours of model run.

dhour,dminut, dday, dmonth, dyear, dlat, dlong	Inactive.
sptime	Inactive.
stname	Inactive.
stnlat	Inactive.
stnlng	Inactive.
stdep	Inactive.
thetap	angle of the grid to the lat-long grid at a designated point.
ip	Integer i-grid points value of output stations.
jp	Integer j-grid points value of output stations.
stni	Real i-grid point value of output stations.
stnj	Real j-grid point value of output stations.
nstns	Number of stations
region	80 character title from the TOPOG.DAT file.
jproj	Flag defining Lambert-conformal projection.
proj	Array of Lambert-conformal map projections.
height	Array of topography.
nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.

Common Blocks: 2DCOMMON

5.2.3.12 Subroutine OMOUT

Calling Sequence: subroutine omout(lu, dfile, region, hour, minut, day, month, year, tstep, wtzone, ntcs, iwind, ntemp, punits, nx, ny, nz)

Data Declaration: real tstep, wtzone, punits
integer nx, ny, nz, hour, minut, day, month year, ntcs, ntemp, iwind
logical lu
char region, dfile

Arguments:

lu	Logical unit number of output file.
dfile	Name of output file (GCOM.DAT).
region	80 character title from the TOPOG.DAT file.
hour	Integer time of current time in model integration.
minut, day, month, year	Time of output.
tstep	Model time step.
wtzone	Wind time zone, in real hours relative to GMT.
ntcs	Number of tidal constituents available at each tidal station.
iwind	Wind forcing flag (0=no wind forcing, 1= wind

	forcing).
ntemp	Inactive.
punits	Multiplier to convert pressure from pascals into mks units.
nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.
nz	Array dimensions in z-direction (=1 for GCOM2D).

Common Blocks: 2DCOMMON

5.2.3.13 Subroutine PHYSIC

PHYSIC calculates the adjustment to the momentum equations due to surface and bottom stress and atmospheric pressure gradient. It is called by the MAIN program.

Calling Sequence: subroutine physic(nx, ny, iwind, dt, chev, rhobar)

Data Declaration: real dt, chev, rhobar
integer nx, ny, iwind

Arguments:

nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.
iwind	Wind forcing flag (0=no wind forcing, 1= wind forcing).
dt	Time step.
chev	Coefficient of horizontal eddy viscosity.
rhobar	Mean density of water.

Common Blocks: 2DCOMMON

5.2.3.14 Subroutine TIDEOBS

This subroutine opens the tidal station data file TCANALS.DAT, and determines which stations are within the model domain. It then carries out tidal height predictions at each station and nudges the modeled sea levels. It is called by the MAIN program.

Calling Sequence: subroutine tideobs(lobs, lthp, tzone lstc, zerr, iassim, tsdt, hour, minut, month, year, nx, ny,dt)

Data Declaration: real dt, tzone, zerr, tsdt
integer nx, ny, iassim, hour, minut, month, year
logical lobs, lthp, lstc

Arguments:

lobs	Logical unit of observation file input.
lthp	Logical unit of tidal prediction output.

tzone	Time zone, in real hours relative to GMT.
lstc	Logical unit of L.
zerr	Mean error between model and tidal stations.
iassim	Assimilation flag (0=no assimilation, 1=assimilation).
tsdt	Time series output time.
hour, minut	
day, month, year	Current time of observations.
nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.
dt	Time step.

Common Blocks: 2DCOMMON

5.2.3.15 Subroutine UVBAR

UVBAR calculates *u* at *v* grid points and *v* at *u* grid points. It is called by the MAIN program.

Calling Sequence: subroutine uvbar(nx, ny)

Data Declaration: integer nx, ny

Arguments:

nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.

Common Blocks: 2DCOMMON

5.2.3.16 Subroutine UVBND

This subroutine applies boundary conditions to velocity components. It is called by the MAIN program.

Calling Sequence: subroutine uvbnd(nx, ny, dt, ibc, nstpts)

Data Declaration: real dt
integer nx, ny, ibc, nstpts

Arguments:

nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.
dt	Model time step.
ibc	Boundary condition flag (set internally to 5).
nstpts	Number of points for nesting in (set internally).

Common Blocks: 2DCOMMON

5.2.3.17 Subroutine UVGRAV

UVGRAV solves the gravity wave terms in the equations of motion. It is called by the MAIN program.

Calling Sequence: subroutine uvgrav(nx, ny,dt)

Data Declaration: real dt
integer nx, ny

Arguments:		
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	dt	Model time step.

Common Blocks: 2DCOMMON

5.2.3.18 Subroutine ZBOUND

ZBOUND inputs nesting heights on boundaries due to tides (from S2.DAT, M2.DAT, etc), coarse mesh model data (from GCOM.NST) or inverse barometer effects (from ATMOS.DAT). It is called by the MAIN program.

Calling Sequence: subroutine zbound(nx, ny,iwind, ipress, itide, nstpts, ibc, dt, rhobar, pbar)

Data Declaration: real dt, rhobar, pbar
integer nx, ny, iwind, ipress, itide, nstpts, ibc

Arguments:		
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	iwind	Wind forcing flag (0=no wind forcing, 1= wind forcing).
	ipress	Pressure flag (0= no atm. pressure, so set to *pbar*)
	itide	Tide forcing flag (0= no tidal forcing, 1= tidal forcing, 2= data assimilation).
	nstpts	Number of points for nesting in (set internally).
	ibc	Boundary condition flag.
	dt	Model time step.
	rhobar	Mean density of water.
	pbar	Mean atmospheric pressure.

Common Blocks: 2DCOMMON

5.2.3.19 Subroutine ZSOLVE

This routine solves the continuity equation. It is called by the MAIN program.

Calling Sequence: subroutine zsolve(nx, ny,dt)

Data Declaration: real dt
integer nx, ny

Arguments:		
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	dt	Model time step.

Common Blocks: 2DCOMMON

5.2.3.20 Subroutine ZVNEST

ZVNEST interpolates the coarse grid GCOM.OUT to fine grid GCOM.NST. It is called by the MAIN program.

Calling Sequence: subroutine zvnest(lu, cgfile,nx, ny,hour minut, day, month, year, dt, spv, ntcsnst)

Data Declaration: real dt,spv
integer nx, ny, hour, minut, day, month, year, ntcsnst
char cgfile
logical lu

Arguments:		
	lu	Logical unit number of coarse grid file interpolated to fine grid.
	cgfile	Coarse grid file name.
	nx	Array dimensions in x-direction.
	ny	Array dimensions in y-direction.
	hour, minut, day, month, year	Nesting time.
	dt	Model time step.
	spv	Special Value.
	ntcsnst	Number of tidal constituents in nesting file.

Common Blocks: 2DCOMMON

5.2.3.21 Subroutine COASTS

COASTS evaluates the wetting and draining at the coastal boundary. It is called by the MAIN program.

Calling Sequence: subroutine coasts(nx, ny, ibf, dt, depmin, depmax, iplot, inund)

Data Declaration: real dt, depmin, depmax
integer nx, ny, ibf, iplot, inund

Arguments:

nx	Array dimensions in x-direction.
ny	Array dimensions in y-direction.
ibf	Bathymetry filtering flag (set internally to 'on').
dt	Model time step.
depmin	Minimum allowable depth in model (set in PARAM).
depmax	Maximum allowable depth in model (set in PARAM).
iplot	Real-time output flag (0=text or 1= graphics to screen)
inund	Inundation flag (1= call inundation algorithm).

Common Blocks: 2DCOMMON

5.2.4 The GCOM System

In addition to GCOM2D, several other modules and programs are necessary to complete the modeling system. These include programs for setting up model grids over specified geographic regions, extraction and conversion programs to provide atmospheric initial and boundary conditions in suitable format for GCOM2D and display programs for viewing model output. These additional programs are summarized in Figure 5.2.4-1 while the key additional elements are described in the following sections.

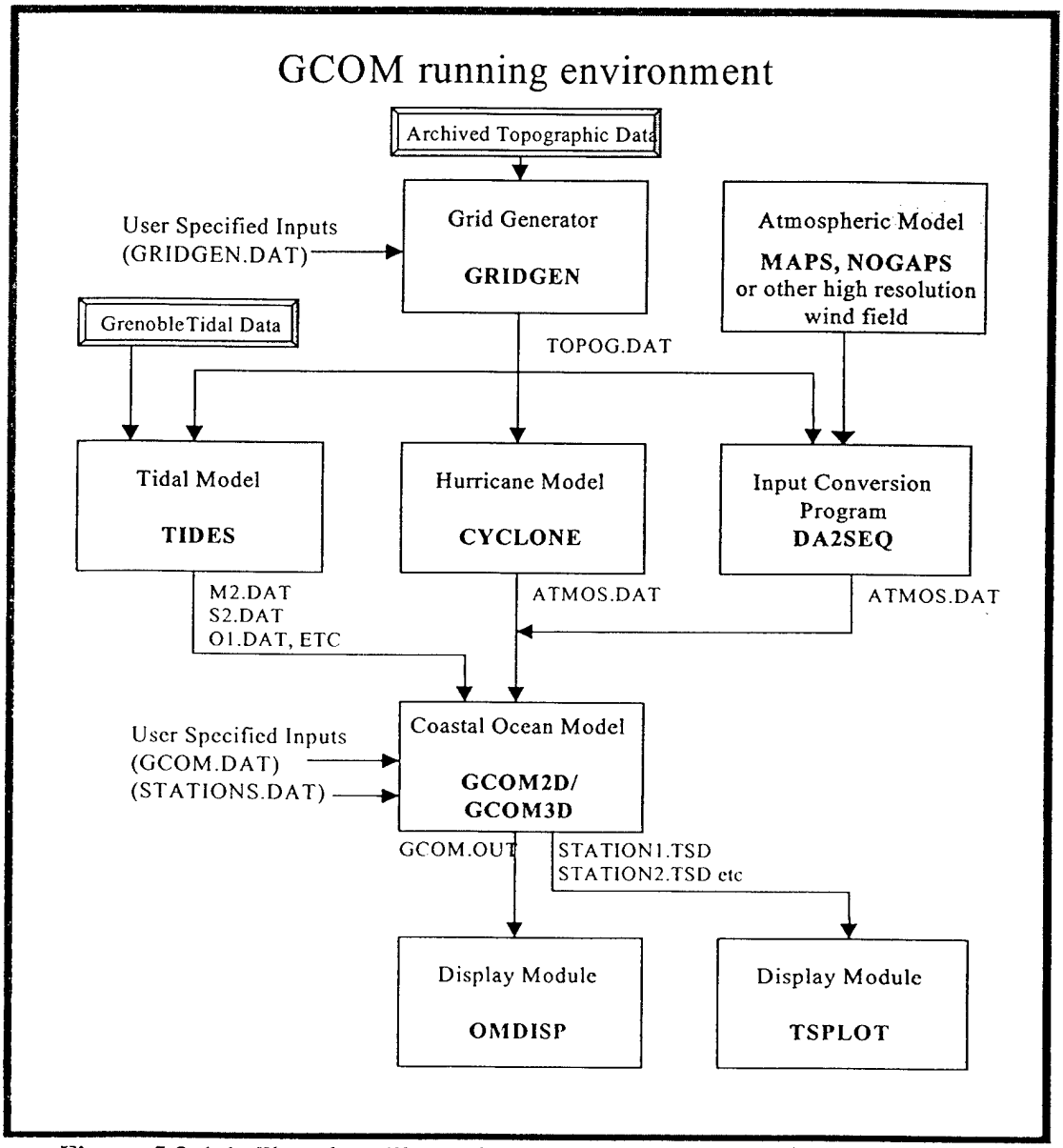

Figure 5.2.4-1: Flowchart illustrating the running of the GCOM2D/3D system.

5.2.4.1 Field Generator

The field generator reads the TOPOG.DAT file, inputs the atmospheric data files (MAPS.000, etc), and horizontally interpolates the fields to the region defined by the TOPOG.DAT parameters. The output fields are written in sequential file format to ATMOS.DAT.

5.2.4.2 Tide Model

The tide model reads the global 30-minute tidal constituent files (GLBM2.30M, GLBS2.30M, GLBO1.30M, GLBK1.30M etc.) derived from the global model, The Finite Element Solutions 95.1.21 (FES 95.1/2.1) (Shum et al., 1997). It then interpolates the data for each constituent to the region defined by the TOPOG.DAT parameters. The output files are named M2.DAT, S2.DAT, O1.DAT, K1.DAT etc. These values may be displayed in the system output model (see sect. 5.2.4.4).

5.2.4.3 Hurricane Model

The primary requirement for modeling storm surges is accurate surface wind and atmospheric pressure fields. There are usually insufficient data to allow a direct analysis of the central region of most tropical cyclones and currently available numerical weather prediction models do not adequately represent the small-scale features. Surface wind speeds and pressures are therefore derived by means of a numerical tropical cyclone model based on the empirical relationships presented by Holland (1980) and similar to the model described in Hubbert et al., (1991). This method of deriving tropical cyclone wind and pressure fields has been tested extensively and has also been used extensively in forecasting offices both in Australia and elsewhere and has been found to provide a good representation of the wind fields in the vicinity of a cyclone.

The pressure P (hPa) at radius r is derived as follows:

$$P = P_c + (P_n - P_c)e^{-(r_m/r)^b} \qquad (5.2.19)$$

where P_c is the central pressure, P_n is the environmental pressure (the climatological mean for the region and month), r_m is the radius of maximum winds and b provides a scaling on the profile shape. The parameter b is empirically defined by

$$b = 1.5 + (980 - P_c)/120 \qquad (5.2.20)$$

The symmetric, gradient-level azimuthal wind component is estimated by

$$v = \left[b\left(\frac{r_m}{r}\right)^b \frac{(P_n - P_c)}{\rho} e^{-(r_m/r)^b} - \frac{r^2 f^2}{4} \right]^{\frac{1}{2}} - \frac{rf}{2} \qquad (5.2.21)$$

where ρ is the air density and f is the Coriolis parameter.

A first order asymmetry is included by adding the cyclone translation to the symmetric field and rotating the field so that the maximum wind is 70° to the left (right in the Northern Hemisphere) of the direction of cyclone motion. The radial wind field is constructed by rotating the flow to a constant inflow angle of 25° outside the radius of maximum winds.

It is important to note that the tropical cyclone model is not expected to represent the full field of synoptic scale features with a high degree of accuracy. The critical aspect for storm surge and wave forcing is that the model parameterizes the mesoscale forcing in the vicinity of the maximum winds reasonably well.

5.2.4.4 Display Modules

There are two options for displaying GCOM2D output. The first of these is OMDISP that plots the simulated variables at designated levels and times during the model simulation and the second, TSPLOT that displays the time varying station output variables (sea level and currents). In addition, the bathymetry may be displayed in a 2-dimensional form using BATHPLOT or in a 3-dimensional form using PLOT3D. Boundary conditions derived from FES95.1/2.1 (Shum et al., 1997) can be displayed with/without IHO tide station data using PLOTIDE.

The organization of subroutines and functions within the GCOM2D/3D model is illustrated in Figure 5.2.4-2.

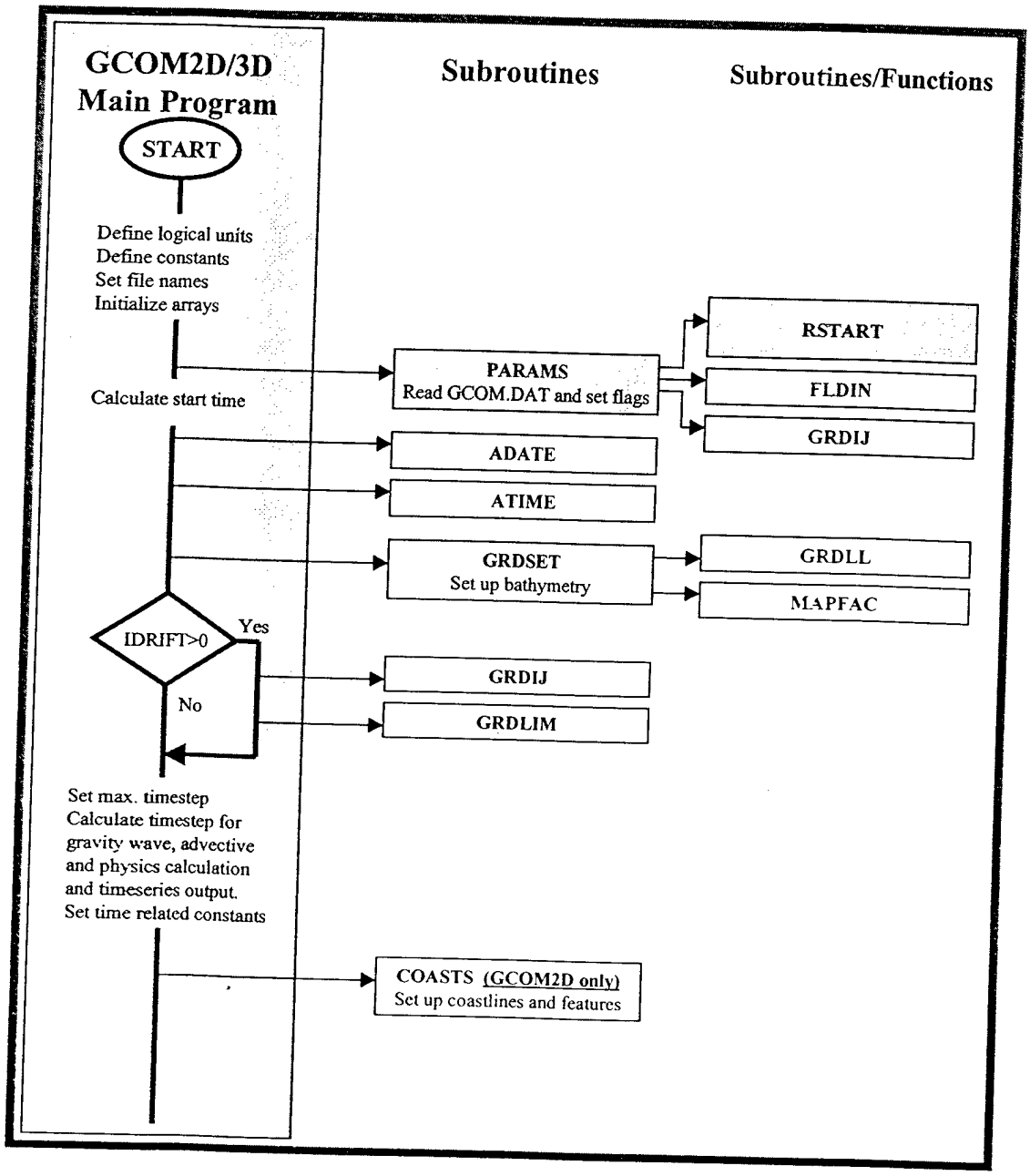

Figure 5.2.4-2: Flowchart governing GCOM2D/3D. Shaded subroutines are found in the GCOM2D/3D programs while unshaded subroutines reside in compiled libraries (see Chapter 5). Programs unique to GCOM2D are indicated with horizontal hatching and programs unique to GCOM3D are indicated with vertical hatching.

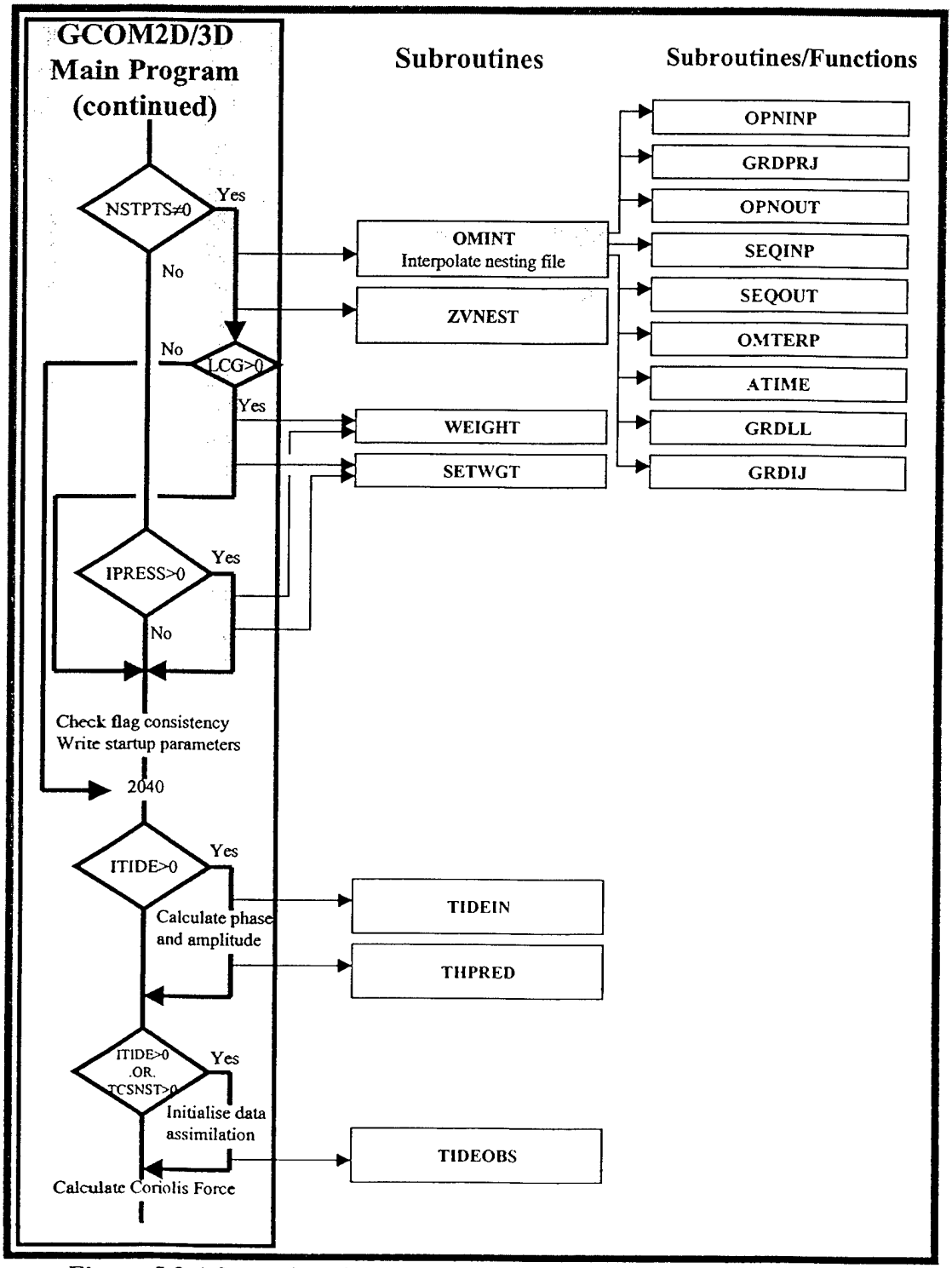

Figure 5.2.4-2: continued.

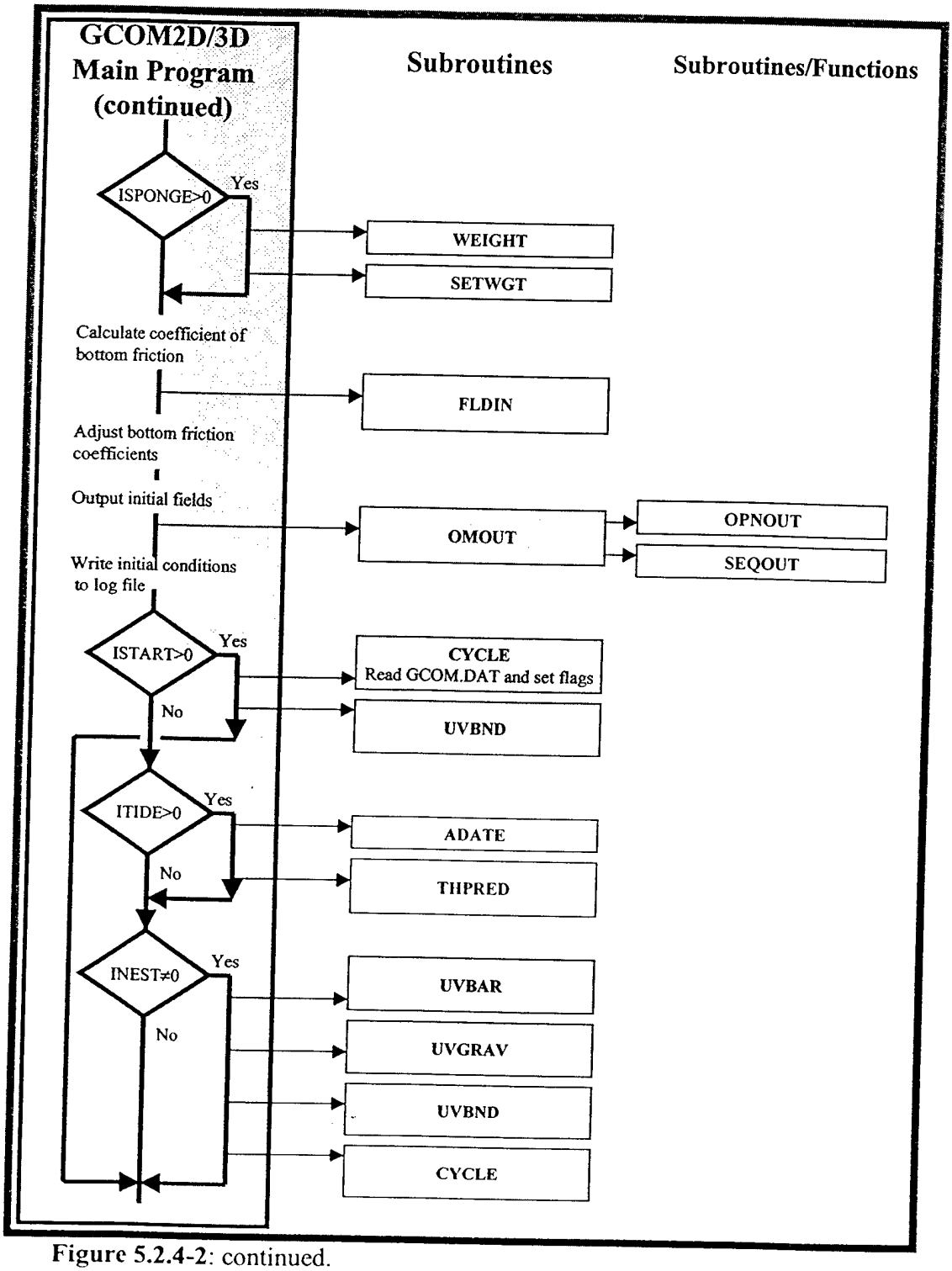

Figure 5.2.4-2: continued.

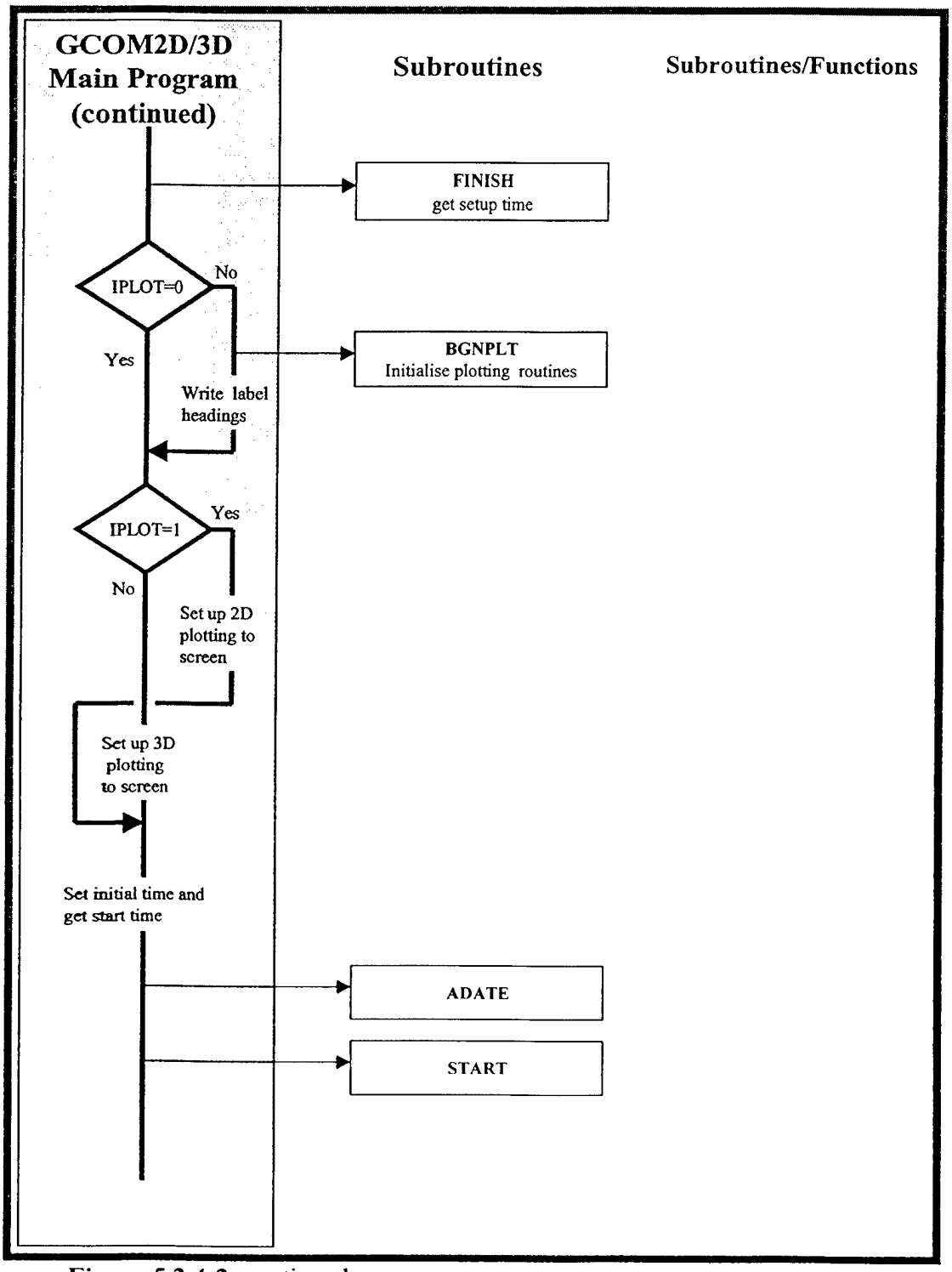

Figure 5.2.4-2: continued.

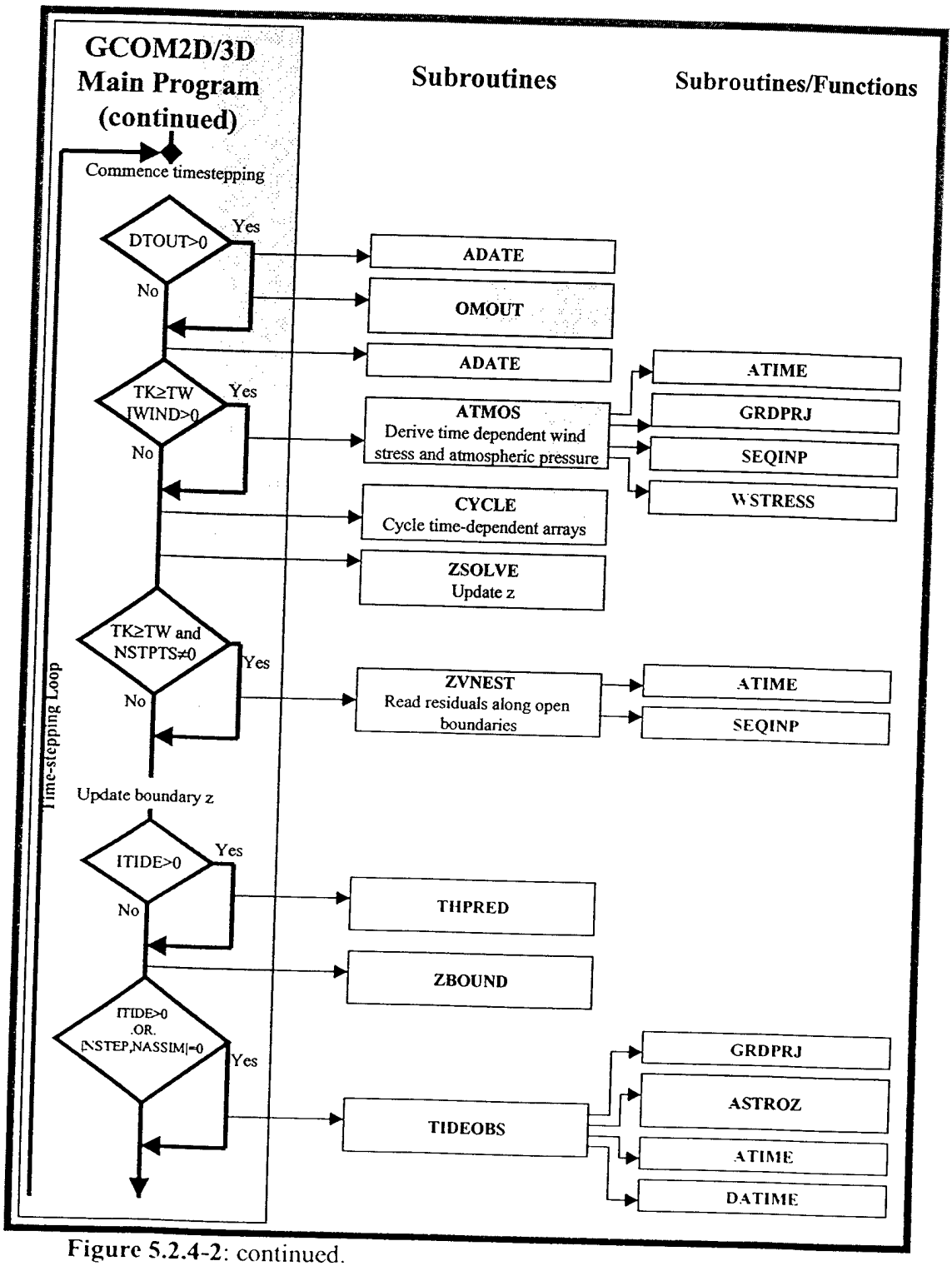

Figure 5.2.4-2: continued.

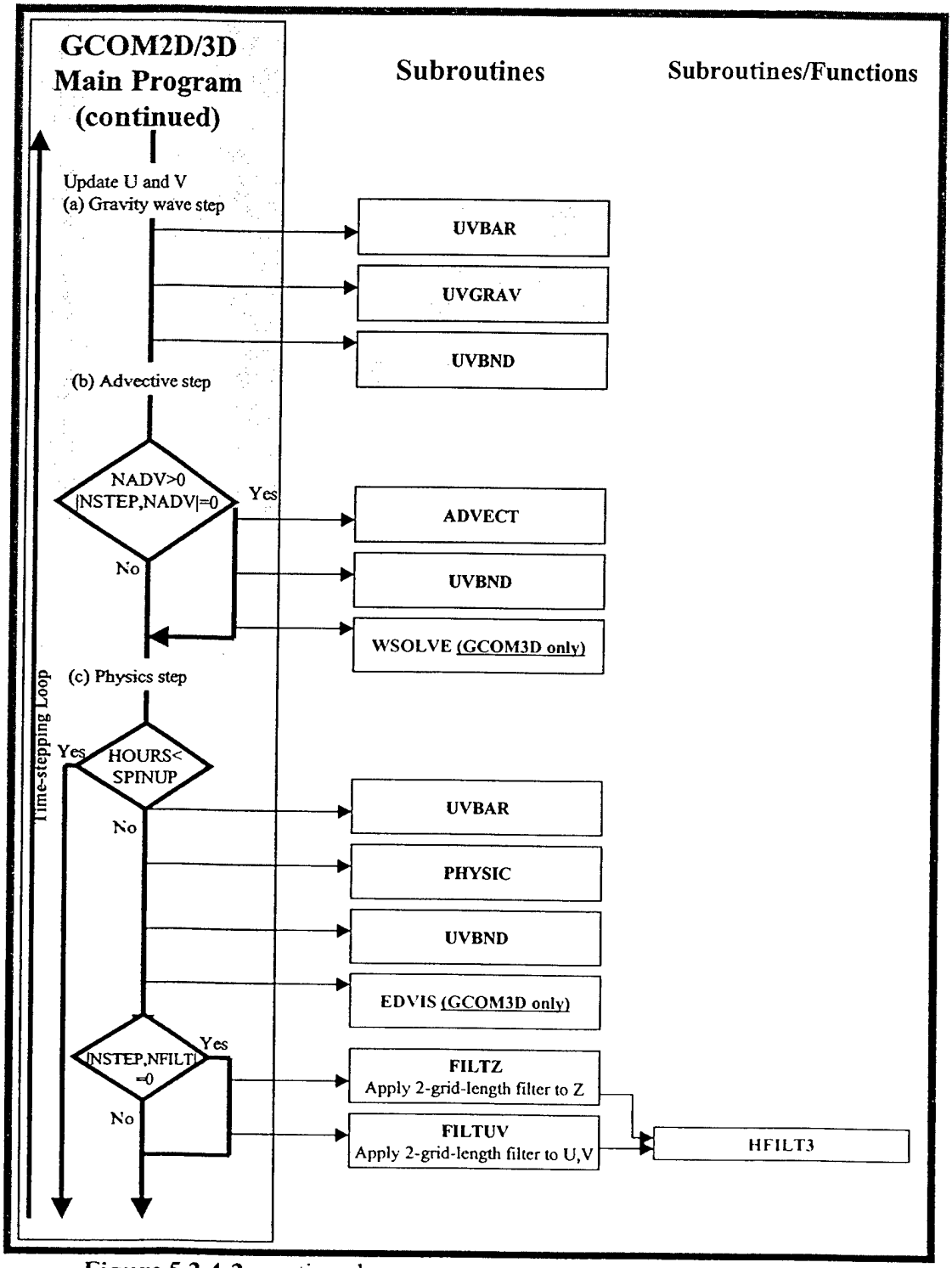

Figure 5.2.4-2: continued.

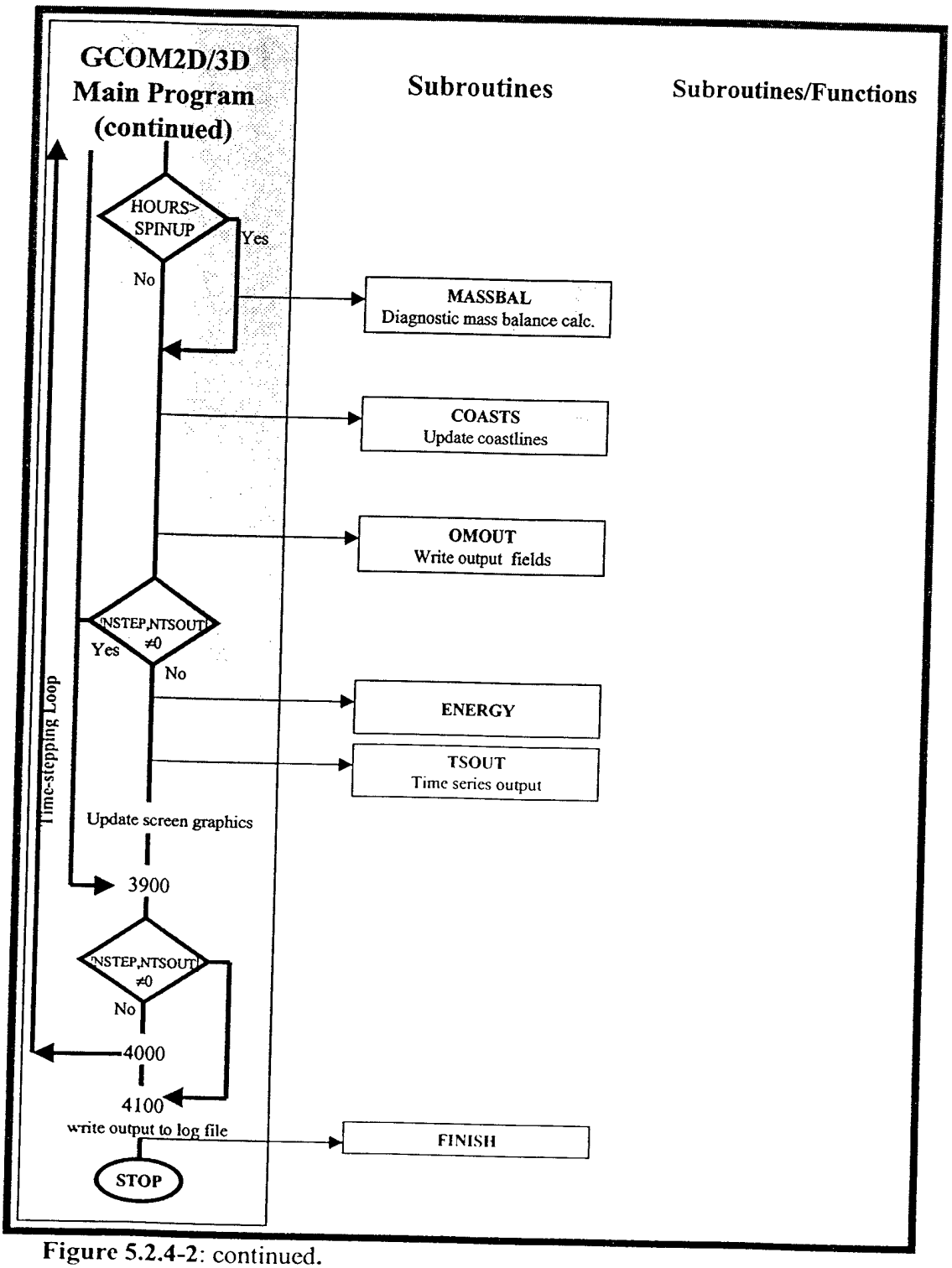

Figure 5.2.4-2: continued.

5.3 CSC GCOM3D

GCOM3D is a 3-dimensional z-coordinate hydrodynamic model that is used for coastal ocean modeling where current structure with depth is required. It can be run in either barotropic mode (no thermal and density variation) or baroclinic mode (temperature and salinity are carried as prognostic model variables). However only the barotropic mode is supplied with this installation.

5.3.1 Constraints and Limitations

See Section 5.2.1.

5.3.2 Logic and Basic Equations

For fully three-dimensional incompressible flow the equations of momentum conservation are:

$$\frac{\partial U_k}{\partial t} = -\frac{\partial}{\partial x}(U_k \cdot U_k) - \frac{\partial}{\partial y}(U_k \cdot V_k) - \frac{\partial}{\partial z}(U_k \cdot W_k) - fV_k - \frac{1}{\rho_k}\frac{\partial P_k}{\partial x} + \frac{1}{\rho_k}\left(\frac{\partial}{\partial x}\tau_k^{xx} + \frac{\partial}{\partial y}\tau_k^{xy} + \frac{\partial}{\partial z}\tau_k^{xz}\right), \tag{5.3.1}$$

$$\frac{\partial V_k}{\partial t} = -\frac{\partial}{\partial x}(U_k \cdot V_k) - \frac{\partial}{\partial y}(V_k \cdot V_k) - \frac{\partial}{\partial z}(W_k \cdot V_k) + fU_k - \frac{1}{\rho_k}\frac{\partial P_k}{\partial y} + \frac{1}{\rho_k}\left(\frac{\partial}{\partial x}\tau_k^{xy} + \frac{\partial}{\partial y}\tau_k^{yy} + \frac{\partial}{\partial z}\tau_k^{yz}\right), \tag{5.3.2}$$

$$\frac{\partial W_k}{\partial t} = -\frac{\partial}{\partial x}(W_k \cdot U_k) - \frac{\partial}{\partial y}(W_k \cdot V_k) - \frac{\partial}{\partial z}(W_k \cdot W_k) - g - \frac{1}{\rho_k}\frac{\partial P_k}{\partial z} + \frac{1}{\rho_k}\left(\frac{\partial}{\partial x}\tau_k^{xz} + \frac{\partial}{\partial y}\tau_k^{yz} + \frac{\partial}{\partial z}\tau_k^{zz}\right), \tag{5.3.3}$$

In coastal ocean flows, there is little vertical motion and so the vertical acceleration and the vertical gradients of the stress terms are negligible compared to g so equation (5.3.3) reduces to the hydrostatic equation,

$$\frac{\partial}{\partial z}P_k + \rho_k \cdot g = 0 \tag{5.3.4}$$

The equation of continuity for an incompressible fluid is:

$$\frac{\partial}{\partial x}U_k + \frac{\partial}{\partial y}V_k + \frac{\partial}{\partial z}W_k = 0 \tag{5.3.5}$$

In order to close this set of equations, the stress terms must be expressed in terms of known variables. The horizontal shearing stresses are usually written in Boussinesq form as;

$$\begin{aligned} \tau^{xz} &= \rho K \frac{\partial U}{\partial z}, & \tau^{yz} &= \rho K \frac{\partial V}{\partial z}, \\ \tau^{xx} &= \rho K_H \frac{\partial U}{\partial x}, & \tau^{xy} &= \rho K_H \frac{\partial V}{\partial x}, \\ \tau^{yx} &= \rho K_H \frac{\partial U}{\partial y} & \tau^{yy} &= \rho K_H \frac{\partial V}{\partial y} \end{aligned} \tag{5.3.6}$$

where $K = K(x, y, z, t)$ is the coefficient of vertical eddy viscosity and K_H is the coefficient of horizontal eddy viscosity and is assumed constant. Equations (5.3.1)-(5.3.6) form the basis for modeling the vertical and horizontal variation of the ocean dynamics.

In general the vertical eddy viscosity K is a function of the flow and not the fluid and there are various levels of approximation which can be adopted. In the barotropic version of GCOM3D, it is assumed that $K = K(U, V, z) = C_K \left[\left(\frac{\partial U}{\partial z}\right)^2 + \left(\frac{\partial V}{\partial z}\right)^2\right]$. This is the formulation adopted by Leendertsee (1970). Another alternative is to assume that K is a constant however. this approach is not found to give realistic vertical mixing. A more sophisticated approach is to solve for K using the turbulent closure models described in Mellor and Yamada (1974, 1982). The specification of surface stress and bottom stress in GCOM3D is identical to that of GCOM2D (see equations 5.2.4-5.2.6).

5.3.3 Data Elements

5.3.3.1 Commonly Used Parameters

File: 3dparams.for

Variable	Type	Description
nk	Integer	Maximum vertical dimension
ni	Integer	Maximum horizontal dimension (x-direction)
nj	Integer	Maximum horizontal dimension (y-direction)

Subroutines

5.3.3.2 Subroutine EDVIS

EDVIS calculates the coefficient of vertical eddy viscosity at each grid point.

Calling Sequence: subroutine edvis (iev, iwind, itide, ny, zk, cev1, cev2, wk, s, cg, cbf)

Data Declaration: real zk, cev1, cev2, wk, s, cg, cbf
integer iev, iwind, itide, ny

Arguments:		
	iev	Eddy viscosity formulation flag (internally set).
	iwind	Wind forcing flag(0= no wind forcing, 1= wind forcing).
	itide	Tide forcing flag (0= no tidal forcing, 1= tidal forcing, 2= data assimilation).
	ny	Number of grid points in the y-direction.
	zk	Constant eddy viscosity (not used).
	cev1	Inactive.
	cev2	Coefficient of vertical eddy viscosity.
	wk	Inactive.
	s	Inactive.
	cg	Inactive.
	cbf	Inactive.

Common Blocks: 3DCOMMON

5.3.3.3 Subroutine WSOLVE

WSOLVE calculates the vertical current velocity.

Calling Sequence: subroutine wsolve(ny)

Data Declaration: integer ny

Arguments:		
	ny	Number of grid points in y-direction.

Common Blocks: 3DCOMMON

Library Programs and Data Files

Libraries are used to store frequently used subroutines that are common to various PCTides programs. This section documents the commonly used sets of library routines.

5.4 Input/Output Routines

(a) Direct Access Files

The MAPS system uses a direct access file format where all model variables at all levels are stored at each given time. There are various subroutines associated with the reading and writing of direct access files and these are stored in the library AMSUBS.FOR. A brief description of these routines follows.

5.4.1 Subroutine DAOPEN

This routine opens the direct access file connected to unit '*lu*'.

Calling Sequence: subroutine daopen (luin,fname,nxy)

Data Declaration: logical luin
integer nxy
char fname

Arguments:

luin	Logical unit number of input file name.
fname	Character variable containing file name.
nxy	Total record length.

5.4.2 Subroutine DACLOS

This subroutine closes the direct access file connected to unit 'LU'.

Calling Sequence: subroutine daclos (lu)

Data Declaration: logical lu

Arguments: lu — Logical unit number of file to be closed.

5.4.3 Subroutine DAPRMS

This subroutine sets up the working elements of the file directory, specifically the grid projection parameters.

Calling Sequence: subroutine daprms(idir,rdir,cdir,isize,nx,ny,nz,jproj.proj)

Data Declaration: integer idir, isize, nx, ny, nz, jproj
real proj, rdir
char cdir

Arguments:

idir	Integer file header information array.
rdir	Real file header information array.
cdir	Character file header information array.
isize	Record length.
nx	The dimension in the x-direction.
ny	The dimension in the y-direction.
nz	The dimension in the vertical direction.
jproj	Projection parameter flag.
proj	Element array of projection parameters.

5.4.4 Subroutine DADIR

This subroutine sets up the working elements of the file directory.

Calling Sequence: subroutine dadir(idir,rdir,cdir,isize,slf,ns,mlf,nm,rlev,nl)

Data Declaration: integer idir, isize, ns, mlf, nm, nl
real rdir, slf, rlev
char cdir

Arguments:

idir	Integer file header information array.
rdir	Real file header information array.
cdir	Character file header information array.
isize	Record length.
slf	Number of single level fields.
ns	Array of single level field names.
mlf	Number of multi-level fields.
nm	Array of multi-level field names.
rlev	Number of levels.
nl	Array of sigma or pressure levels.

5.4.5 Subroutine DALEVS

This routine extracts the levels information from the directory.

Calling Sequence: subroutine dalevs(idir,rdir,cdir,isize,rlev,nl)

Data Declaration: integer idir, isize, nl
real rdir, rlev
char cdir

Arguments: See DADIR

5.4.6 Subroutine DARDDR

This routine obtains the directory from file record 1.

Calling Sequence: subroutine darddr (lu,idir,rdir,cdir,isize)

Data Declaration: logical lu
integer idir, isize
real rdir
char cdir

Arguments: See DADIR

5.4.7 Subroutine DAWRDR

The purpose of this subroutine is to write the directory to file record 1.

Calling Sequence: subroutine dawrdr (lu,idir,rdir,cdir,isize)

Data Declaration: logical lu
real rdir
integer idir,isize
char cdir

Arguments: See DADIR

5.4.8 Subroutine DAINP

This routine reads the field as specified by name and level. If IRET=0, the data was extracted successfully, IRET=1, the field or level was invalid or not present, IRET=2, there was an error while reading the file.

Calling Sequence: subroutine dainp(lu,idir,rdir,cdir,name,rlev,array,ni,nx,ny,iret)

Data Declaration: logical lu
real rdir, rlev, array
integer idir, ni, nx, ny, iret
char cdir, name

Arguments:	lu	Logical unit number of input file.
	idir	Integer file header information array.
	rdir	Real file header information array.
	cdir	Character file header information array.
	name	4 Character string containing name of field to be read in.
	rlev	1-D array containing vertical levels.
	array	2-D array to be read in.
	ni	The first dimension of ARRAY.
	nx	The number of elements in the 1st dimension of ARRAY.
	ny	The number of elements in the 2nd dimension of ARRAY.
	iret	Error flag (=0 for no error, =1 if field was not present).

5.4.9 Subroutine DAOUT

This routine writes the field as specified by name and level. If IRET=0, the data was extracted successfully, IRET=1, the field or level was invalid or not present, IRET=2, there was an error while reading the file.

Calling Sequence: subroutine daout(lu,idir,rdir,cdir,name,rlev,array,ni,nx,ny,iret)

Data Declaration: logical lu
real rdir, rlev, array
integer idir, ni, nx, ny, iret
char name, cdir

Arguments: See DAINP

(b) Sequential Files

GCOM2D uses a sequential file format for storage of output data. The subroutines associated with sequential file creation and access are stored in STSUBS.FOR

5.4.10 Subroutine OPNINP

This routine opens a sequential file and reads the parameter arrays.

Calling Sequence: subroutine opninp(lunit,fname,title,iparam,rparam,nparam)

Data Declaration: logical lunit
char fname, title
integer iparam, nparam
real rparam

Arguments:		
	lunit	Logical unit of sequential file.
	fname	Character variable containing file name.
	title	80 character title.
	iparam	Integer file header information.
	rparam	Real file.
	nparam	Dimension of IPARAM and RPARAM.

5.4.11 Subroutine SEQINP

This routine reads the fields from a sequential file.

Calling Sequence: subroutine seqinp(lunit,var,ni,nx,ny,k,title,hour,minut,day,month, year)

Data Declaration: logical lunit
real var
integer ni, nx, ny, k, hour, minut, day, month, year
char title

Arguments:		
	lunit	Logical unit number of the sequential file.
	var	Array to be read in.
	ni	The first dimension of VAR.
	nx	The number of elements in the 1st dimension of VAR.
	ny	The number of elements in the 2nd dimension of VAR.
	k	The vertical level.
	title	80 character title for labeling.
	hour, minut, day, month, year	The time to which the input array corresponds.

5.4.12 Subroutine OPNOUT

This routine opens a sequential file and writes the parameter arrays.

Calling Sequence: subroutine opnout(lunit,fname,title,iparam,rparam,nparam)

Data Declaration: logical lunit
real rparam
integer iparam, nparam
char fname, title

Arguments: See OPNINP

5.4.13 Subroutine SEQOUT

This routine writes the fields to a sequential file.

Calling Sequence: subroutine seqout(lunit,var,ni,nx,ny,k,title,hour,minut,day,month, year)

Data Declaration: logical lunit
real var, title
integer ni, nx, ny, k, hour, minut, day, month, year

Arguments: See SEQINP

(c) Time Series Files

A standard ASCII format is used for storing station information in both the atmospheric and ocean models. Model output written in this format can be subsequently plotted using the display program TSPLOT.

5.4.14 Subroutine TSDOUT

This subroutine writes a standard time series file. The five standard fields stored in atmospheric time series files are station pressure, wind speed, wind direction, temperature and rainfall. The standard three fields stored in ocean model time series files are sea surface height, current speed, current direction. The remaining two columns of data are nominally saved for ocean temperature and salinity which is not relevant in the current application.

Calling Sequence: subroutine tsdout (lu,stname,stlat,stlong,tzone,depth,ihour, iminut,iday,imonth,iyear, spd,dir,z,temp,salin,npts,inter,source)

Data Declaration: real stlat, stlong, tzone, depth, spd, dir, z, temp, salin
integer ihour, iminut, iyear, iday, imonth, npts, inter
char stname, source
logical lu

Arguments:

lu	Logical unit number of time series output file.
stname	80 character string containing station name.
stlat	Station latitude.
stlong	Station longitude.
tzone	Wind time zone, in real hours relative to GMT.
depth	Station depth above (-ve) or below (+ve) mean sea level.
ihour, iminut, iday, imonth, iyear	The time that the output arrays correspond to.
spd	Current speed array.
dir	Current direction array.
z	Sea level array.
temp	Inactive.
salin	Inactive.
npts	Dimension of output arrays.
inter	Timestep of output in minutes.
source	Character string describing source of time series output.

5.4.15 Subroutine TSDINP

The purpose of this subroutine is to read a standard time series file.

Calling Sequence: subroutine tsdinp(lu,stname,stlat,stlong,tzone,depth,
ihour,iminut,iday,imonth,iyear,
spd,dir,z,temp,salin,nobs,npts,inter,source)

Data Declaration: real stlat, stlong, tzone, depth, spd, dir, z, temp, salin
integer ihour, iminut, iyear, iday, imonth, nobs, npts, inter
char stname, source
logical lu

Arguments: See TSDOUT

(d) Topography Files

A standard ASCII format is used for writing the topography and bathymetry of a model grid. Output written in this format can be subsequently plotted using the display program TOPLOT.

5.4.16 Subroutine TOPOUT

This subroutine writes a standard topography file. The units are in meters and the zero elevation denotes mean sea level. Topography is positive and bathymetry is negative.

Calling Sequence: subroutine topout(lu,title,jproj,proj,height,ni,nx,ny)

Data Declaration: real proj, height
integer jproj,ni, nx, ny
char title
logical lu

Arguments:		
	lu	Logical unit number of input file.
	title	80 character string for description of file.
	jproj	Projection parameter flag.
	proj	7 element array of projection parameters.
	height	Array of heights.
	ni	The first dimension of HEIGHT.
	nx	The number of elements in the 1st dimension of HEIGHT.
	ny	The number of elements in the 2nd dimension of HEIGHT.

5.4.17 Subroutine FLDIN

The purpose of this subroutine is to read a standard topography file.

Calling Sequence: subroutine fldin(lu,region,jproj,proj,array,ni,nx,ny)

Data Declaration: real proj, array
integer jproj, ni, nx, ny
char region
logical lu

Arguments:		
	lu	Logical unit number of input file.
	region	80 character string for description of file.
	jproj	Projection parameter flag .
	proj	7 element array of projection parameters.
	array	Array of heights.
	ni	The first dimension of ARRAY.
	nx	The number of elements in the 1st dimension of ARRAY.
	ny	The number of elements in the 2nd dimension of ARRAY.

5.5 Projection Routines

In the PCTides system, both MAPS and GCOM2/3D make use of a standard Cartesian grid system for the solution of the equations of motion. The curvature of the earth is taken into account by the use of a Lambert conformal map projection that consists of projecting the surface of the earth onto a right circular cone and flattening it onto a plane surface. This process results in a map scale factor *m*, which is defined by the ratio of the distance on the grid to the corresponding distance on the earth's surface. The subroutines associated with grid projection are described as follows.

5.5.1 Subroutine GRDPRJ

This routine initializes the grid projection. A standard Lambert conformal map projection is used (see for example, Haltiner and Williams, 1980). For a Lambert conformal map projection, JPROJ=3. PROJ is a real array of dimension 7 that contains the map projection information as follows:

Calling Sequence: subroutine grdprj(jproj,proj)

Data Declaration: real proj
integer jproj

Arguments:		
	jproj	Projection parameter flag .
	proj	7 element array of projection parameters.
	PROJ(1)	STLAT1: First standard latitude.
	PROJ(2)	ANML: Normal longitude.
	PROJ(3)	STLAT2: Second standard latitude.
	PROJ(4)	DU: Number of grid-points from the southwest corner of grid to ANML.
	PROJ(5)	TANL: Latitude that forms a tangent with ANML at southernmost point on grid.
	PROJ(6)	NY: Number of grid rows.
	PROJ(7)	STDXY: Standard grid spacing in kms.

5.5.2 Subroutine GRDLL

The purpose of this subroutine is to return the (i,j) coordinates for a specified latitude and longitude.

Calling Sequence: entry grdll(jproj,rlat,rlong,ri,rj,theta)

Data Declaration: real rlat, rlong, ri, rj, theta
integer jproj

Arguments:		
	jproj	Projection parameter flag (set to 3 for Lambert conformal).
	rlat	Latitude (supplied).
	rlong	Longitude (supplied).
	ri	The i-coordinate (returned).
	rj	The j-coordinate (returned).
	theta	Inactive.

5.5.3 Subroutine GRDIJ

The purpose of this subroutine is to return the latitude and longitude for a specified (i,j) point.

Calling Sequence: entry grdij(jproj,rlat,rlong,ri,rj,theta)

Data Declaration: real rlat, rlong, ri, rj, theta
integer jproj

Arguments: See GRDLL

5.5.4 Subroutine GRDLIM

The purpose of this subroutine is to calculate the latitude and longitude limits of a grid.

Calling Sequence: subroutine grdlim(jproj,nx,ny,slat,nlat,wlong,elong,region)

Data Declaration: real slat, wlong, elong
integer jproj, nx, ny, nlat
char region

Arguments:		
	jproj	Projection parameter flag .
	nx	Number of grid-points in x-direction.
	ny	Number of grid-points in y-direction.
	slat	Southern latitude limit of grid.
	nlat	Northern latitude limit of grid.
	wlong	Western latitude limit of grid.
	elong	Eastern latitude limit of grid.
	region	80 character string for description of file.

5.6 Date/Time Routines

Date and time routines are used to convert between date and time specified on input files and an invariant reference time used by the model. The reference time used by the PCTides models is the 1st January 1900.

5.6.1 Subroutine ATIME

The purpose of this subroutine is to return a double precision number of hours since 1st January 1900 for a supplied hour (hh), minute (mm), day (DD), month (MM) and year (YYYY).

Calling Sequence: subroutine atime(hour,minut,day,month,year,t)

Data Declaration: real t
integer hour, minut, day, month, year

Arguments:		
	hour	Hour (supplied).
	minut	Minute (supplied).
	day	Day (supplied).
	month	Month (supplied).
	year	Year (supplied).
	t	Number of hours since origin.

5.6.2 Subroutine ADATE

The purpose of this subroutine is to return the hour (hh), minute (mm), day (DD), month (MM) and year (YYYY) when supplied with a double precision number of hours since 1st January 1900.

Calling Sequence: entry adate(hour,minut,day,month,year,t)

Data Declaration: real t
integer hour, minut, day, month, year

Arguments: See ATIME

5.6.3 Subroutine ATIME2

The purpose of this subroutine is to return a double precision number of hours since 1st January 1900 when supplied with the date in (YYYYMMDD) format and the time in (hhmm) format.

Calling Sequence: subroutine atime2(date,time,t)

Data Declaration: real date, time, t

Arguments:		
	date	YYYYMMDD.
	time	HHMM.
	t	Time since origin.

5.6.4 Subroutine ADATE2

The purpose of this subroutine is to return the date in (YYYYMMHH) format and time in (hhmm), format when supplied with a double precision number of hours since 1st January 1900.

Calling Sequence: entry adate2(date,time,t)

Data Declaration: real date, time, t

Arguments: See ATIME2

5.7 Filtering Routines

It is common practice in finite-difference models of the atmosphere and ocean to apply numerical filters to the model simulated variables (Haltiner and Williams, 1980). This is to reduce numerical errors and computational instabilities that may lead to a spurious growth of short waves that may mask the actual numerical simulation or, in severe cases, could cause catastrophic instability. In GCOM2D, the following five-point "low-pass" filter (Shapiro, 1970) is applied to the model variables in each grid direction approximately once every two to ten physics time-steps;

$$\phi_i^* = a\phi_{i-2} + b\phi_{i-1} + c\phi_i + b\phi_{i-1} + a\phi_{i-2}$$

where ϕ represents the particular model variable being filtered and ϕ^*, the filtered variable. The coefficients *a, b*, and *c* are –0.0625, 0.25 and 0.625, respectively. The filter specifically targets two-grid-interval and some sub-grid-interval waves, damps all other waves except for the infinitely long wave. Wave phases remain unchanged by the filtering process.

5.7.1 Subroutine HFILT

HFILT filters the entire array that is supplied to it.

Calling Sequence: subroutine hfilt(array,ni,nx,ny,r1)

Data Declaration: real array, r1
integer ni, nx, ny

Arguments:	array	Array of values to be filtered.
	ni	The first dimension of ARRAY.
	nx	The number elements in the 1st dimensions of ARRAY.
	ny	The number elements in the 2nd dimensions of ARRAY.
	r1	Inactive.

5.7.2 Subroutine HFILT3

HFILT3 is used for filtering ocean model fields that are interrupted by irregular coastlines. Land points are defined by special values.

Calling Sequence: subroutine hfilt3(array,ni,nx,ny,r1,spval)

Data Declaration: real array, r1, spval
integer ni,nx, ny

Arguments:		
	array	Array of values to be filtered.
	ni	The first dimension of ARRAY.
	nx	The number elements in the 1st dimensions of ARRAY.
	ny	The number elements in the 2nd dimensions of ARRAY.
	rl	Inactive.
	spval	special value used to mask out elements in an array.

5.8 Miscellaneous MAPS Routines

There are four subroutines pertaining to the solution of the semi-implicit equations in MAPS that reside in the AMSUBS library.

5.8.1 Subroutine MATINV

This routine inverts a matrix using Gauss-Jordon pivotal elimination.

Calling Sequence: subroutine matinv(a,b,nmax,n,l,d,irror)

Data Declaration: real a, b, d
integer nmax, n, l, irror

Arguments:		
	a	N×N matrix to be inverted on entry, inverse on exit.
	b	N×N matrix = RHS of the equation on entry, solution on exit.
	nmax	The first dimension of the arrays A and B.
	n	The number of elements stored in A and B.
	l	If L>0, solutions are given; L=0, inverse is given; L<0, both are given
	d	Determinant of matrix.
	irror	=1 if matrix is singular =0 if matrix is inverted successfully.

5.8.2 Subroutine OPTMUM

The purpose of this subroutine is to optimize the over-relaxation coefficient for the routine LIEBH.

Calling Sequence: subroutine optmum(idim,jdim,alfa,sigma)

Data Declaration: real alfa, sigma
integer idim, jdim

Arguments:		
	idim	NX-1.
	jdim	NJ-1.
	alfa	Relaxation coefficient calculated in OPTMUM.
	sigma	Input required for calculation of relaxation coefficient.

5.8.3 Subroutine EIGENP

The purpose of this subroutine is to find the eigenvalues and the eigenvectors of a real general matrix of order N.

Calling Sequence: subroutine eigenp(a,vecr,veci,evr,evi,nm,n,indic)

Data Declaration: real a, vecr, veci, evr, evi
integer nm, n, indic

Arguments:		
	a	Matrix of N rows and N columns.
	vecr	N×N matrix of real eigenvectors calculated in the routine.
	veci	N×N matrix of imaginary eigenvectors.
	evr	Vector of N dimension containing the real eigenvalues.
	evi	Vector of N dimension containing the imaginary eigenvalues.
	nm	The first dimension of the arrays A, VECR, VECI.
	n	The number of elements stored in A, VECR, VECI, EVR, EVI.
	indic	=0 when no eigenvalues or vectors are found =1 when eigenvalues are found but eigenvectors are not found =2 when both eigenvalues and eigenvectors are found.

5.8.4 Subroutine LIEBH

The purpose of this subroutine is to solve the Helmholtz equations.

Calling Sequence: subroutine liebh(array,sol,emsqi,alf,itns,sigma,il,jl,nx,ny,nz)

Data Declaration: real array, sol, emsqi, alf, sigma
integer itns, il, jl, nx, ny, nz

Arguments:		
	array	IL×JL×NZ array of input to subroutine.
	sol	IL×JL×NZ array containing solution of Helmholtz equations.
	emsqi	Inverse of map factor squared.
	alf	Vector of NZ relaxation coefficients.
	itns	Vector of No. of iterations required for solution at each level.
	sigma	Vector of NZ eigenvalues multiplied by Δs^2.
	il	The first dimension of ARRAY and SOL.
	jl	The second dimension of ARRAY and SOL.
	nx	No. of elements in first dimension of ARRAY and SOL.
	ny	No. of elements in second dimension of ARRAY and SOL.
	nz	The third dimension of ARRAY and SOL. No. of vert. levels.

5.9 Miscellaneous GCOM Routines

There are several subroutines called by GCOM2/3D that are stored in the library STSUBS.FOR.

5.9.1 Subroutine WEIGHT

The purpose of this subroutine is to calculate the boundary weight factors for nesting and application of sponge layers.

Calling Sequence: subroutine weight(npts,linear,wgt,ni,nx,ny)

Data Declaration: real wgt
integer npts, linear, ni, nx, ny

Arguments:		
	npts	Number of nesting points.
	linear	Interpolation flag (internally set to 1).
	wgt	Array of nesting weights.
	ni	The first dimension of *depz*.
	nx	The number of elements stored in the 1st dimension of *depz*.
	ny	The number of elements stored in the 2nd dimension of *depz*.

5.9.2 Subroutine SETWGT

The purpose of this subroutine is to set the nesting weights.

Calling Sequence: subroutine setwgt(wgt,depz,ni,nx,ny,npts)

Data Declaration: real wgt, depz
integer ni, nx, ny, npts

Arguments:		
	wgt	Array of nesting weights.
	depz	Array of water depths relative to mean sea level.
	ni	The first dimension of *depz* and *wgt*.
	nx	The number of grid points in x-direction.
	ny	The number of grid points in y-direction.
	npts	Number of nesting points.

5.9.3 Subroutine START

The purpose of this subroutine is to calculate the start time from the system clock.

5.9.4 Subroutine THPRED

The purpose of this subroutine is to read astronomical constants for each tidal constituent and calculate resulting tidal height.

Calling Sequence: subroutine thpred(lstc,con,ntcmax,ntcs,amp,phase,chp,ch,rlat, tdhgt,ni,nx,ny,hour,minut,day,month,year,dt,tzone)

Data Declaration: logical lstc
real con, amp, phase, chp, ch, rlat, tdhgt, dt, tzone
integer ntcmax, ntcs, ni, nx, ny, hour, minut, day, month, year

Arguments:		
	lstc	Logical unit which reads the standard tidal constituent data from STCONTHP.DAT.
	con	Tidal constituents array; CON(NTCMAX,NOBS).
	ntcmax	Maximum number of tidal constituents.
	tcs	Number of tidal constituents available at each tidal station.
	amp	Tidal amplitudes read in from M2.DAT, S2.DAT etc.
	phase	Tidal phases read in from M2.DAT etc.
	chp	Working array dimensioned (NTCMAX,NX,NY).
	ch	As above.
	rlat	Station latitude.
	tdhgt	Predicted tidal height.
	ni	First dimension of AMP, PHASE and TDHGT.
	nx	The number of elements in the 1st dimension of AMP, etc.
	ny	The number of elements in the 2nd dimension of AMP, etc.
	hour, minut, day, month, year	Time of required tidal height prediction.
	dt	Model time-step.
	tzone	Wind time zone, in real hours relative to GMT.

5.10 Data Files for GCOM2D/3D and MAPS

In order for MAPS and GCOM2/3D to be globally relocatable there needs to be global climatological and environmental data files to establish basic grid and model fields and parameters. These data files are summarized below.

5.10.1 Topographic/Bathymetric Data

Topographic/bathymetric data are stored on latitude/longitude grids in a series of direct access files, all with the file extension ".DA". All values are stored as INTEGER*2 and the direct access file structure is as follows:

Record 1: Southern latitude limit
Record 2: Northern latitude limit
Record 3: Western longitude limit (0-360)
Record 4: Eastern longitude limit. (0-360)
Record 5: Resolution in minutes x 1000
Record 6: Topography – column 1, row 1
Record 7: Topography – column 2, row 1
Etc.

5.10.1.1 Subroutine GETOP

This routine returns the topography VALUE at the point (RLAT, RLONG) from the highest resolution ".DA" file covering the point. GETOP is in the library file DASUBS.FOR.

Calling Sequence: getop(rlat,rlong,value)

Data Declaration: real rlat, rlong, value

Arguments:		
	rlat	Station latitude.
	rlong	Station Longitude.
	value	Topography value.

5.10.2 Climatological Data

MAPS also requires some global climatological data for roughness length, albedo and soil moisture. Roughness length, albedo and monthly soil moisture values are stored in INTEGER*2 format in the direct access files GLROUGHL.DA, GLALBEDO.DA and GLSOILMV.JAN, GLSOILMV.FEB etc. The direct access file structure is as follows:

Record 1: Southern latitude limit
Record 2: Northern latitude limit
Record 3: Western longitude limit (0-360)
Record 4: Eastern longitude limit. (0-360)
Record 5: Resolution in minutes
Record 6: Factor to multiply data by
Record 7: Value – column 1, row 1
Record 8: Value – column 2, row 1
Etc.

5.10.2.1 Subroutine DA2DREAD

This routine reads the file FNAME, on logical unit LU, and returns the VALUE at the point (RLAT,RLONG). DA2DREAD is in the library file DASUBS.FOR.

Calling Sequence: da2dread (lu,fname,rlat,rlong,value)

Data Declaration: logical lu
char fname
real rlat, rlong, value

Arguments:		
	lu	Logical unit number of the model run log file.
	fname	Character variable containing file name.
	rlat	Station latitude.
	rlong	Station Longitude.
	value	Value.

The program DAFILE.EXE provides the ability to convert these files between an ASCII format and there direct access binary format. The program recognizes the extension ".DA" as the direct access file and the extension ".ASC" as the ASCII file.

5.10.3 Tidal Data

GCOM2/3D also require global tidal files from the FES95.1/2.1 model to derive boundary tidal forcing. These 30 minute resolution data are stored in the ASCII files GLBM2.30M, GLBS2.30M, etc. These routines are in the library file STSUBS.FOR.

5.10.3.1 Subroutine GETIDE

This routine reads these ASCII files and returns the amplitude, AMP, phase, PHASE, and period, PERIOD, of the tidal constituent, CON, at the point (RLAT,RLONG), in time zone, TZONE. GETIDE is in the library file DASUBS.FOR.

Calling Sequence: getide(rlat,rlong,con,amp,phase,period,tzone)

Data Declaration: real rlat, rlong, con, amp, phase, period, tzone

Arguments:		
	rlat	Station latitude.
	rlong	Station Longitude.
	con	Tidal constituent.
	amp	Amplitude.
	phase	Phase.
	period	Period.
	tzone	Time zone.

5.10.3.2 Subroutine TIDEOUT

The purpose of this routine is to write an ASCII tidal file on logical unit, LU, for the region, REGION, projection parameters, JPROJ, PROJ(7), tidal constituent, CON, with period,

PERIOD, in time zone, TZONE, and amplitude, AMP, and phase, PHASE, arrays with dimensions, NX, NY.

Calling Sequence: tidout(lu,region,jproj,proj,con,period,tzone,amp,phase,ni,nx,ny)

Data Declaration: logical lu
char region
integer jproj, ni, nx, ny
real proj, con, period, tzone, amp, phase

Arguments:		
	lu	Logical unit number of the model run log file.
	region	80 character title from the TOPOG.DAT file.
	jproj	Flag defining Lambert-conformal projection.
	proj	Array of Lambert-conformal map projections.
	con	Tidal constituent.
	period	Period.
	tzone	Time zone.
	amp	Amplitude.
	phase	Phase.
	ni	Maximum horizontal dimension (x-direction).
	nx	The number of grid points in x-direction.
	ny	The number of grid points in y-direction.

5.10.3.3 Subroutine TIDEIN

The purpose of this subroutine is to read an ASCII tidal file.

Calling Sequence: tidein(lu,region,jproj,proj,con,period,tzone,amp,phase,ni,nx,ny)

Data Declaration: logical lu
real region,proj, con, period, tzone, amp, phase. ni, nx,ny
integer jproj

Arguments:		
	lu	Logical unit number of the model run log file.
	region	80 character title from the TOPOG.DAT file.
	jproj	Flag defining Lambert-conformal projection.
	proj	Array of Lambert-conformal map projections.
	con	Tidal constituent.
	period	Period.
	tzone	Time zone.
	amp	Amplitude.
	phase	Phase.
	ni	Maximum horizontal dimension (x-direction).
	nx	The number of grid points in x-direction.
	ny	The number of grid points in y-direction.

6.0 REQUIREMENTS TRACEABILITY

The Software Test Description (STD) includes eight test runs that encompass all CSCI routines (SDD) and requirements (SRS).

SRS paragraph numbers:

SDD Paragraph numbers:	3.1.1	3.1.2	3.1.3	3.1.4	3.1.5	3.1.1A	3.1.2A	3.1.3A	3.1.4A
5.1		x	x					x	
5.1.2								x	
5.1.3								x	
5.1.4						x	x	x	x
5.2				x					
5.2.2				x					
5.2.3			x	x					
5.2.4	x	x		x	x				x
5.3				x					
5.3.2				x					
5.3.3				x					
5.4	x					x			
5.5	x								
5.6	x								
5.7	x								
5.8								x	
5.9				x					
5.10.1	x					x			
5.10.2		x					x		
5.10.3	x								

7.0 NOTES

7.1 Acronyms and Abbreviations

ASA	Applied Sciences Associates
CSCI	Computer Software Configuration Item
CSC	Computer Software Component
FES	Finite Element Solution
GCOM2D	Coastal Ocean Model 2-D
GCOM3D	Coastal Ocean Model 3-D
GMT	Generic Mapping Tool
MAPS	Mesoscale Atmospheric Prediction System
NOGAPS	Navy Operational global Atmospheric Prediction System
NRL	Naval Research Laboratory
OAML	Oceanographic and Atmospheric Master Library
PC	Personal Computer
PCTides	Globally Relocatable Navy Tide/Atmospheric Modeling System
PSI	Planning Systems, Incorporated
SDD	Software Design Description
SRS	Software Requirements Specification
SSC	Stennis Space Center
STD	Software Test Description
UNIX	Workstation Operating System

8.0 Appendix I. FORTRAN Common Blocks

CSC Maps COMMON Blocks

COMMON / DOUBLE/

This common block contains all double precision arrays (both two-dimensional and three-dimensional).

Variable	Type	Description
sol (ni, nj, nk), array (ni, nj, nk)	Real	Arrays used for solution of Helmholtz equations; *array* contains the forcing function.
emsqi (ni,nj)	Real	Inverse of the square of the map factor, m^{-2}.
psm (ni, nj), ps(ni,nj), psp(ni,nj)	Real	Surface pressure at the previous, current, and next time steps.
ainv(nk, nk), fcainv (nk, nk), feainv (nk,nk), h (nk,nk), fmat (nk,nk), vecr (nk,nk), vecr1 (nk, nk), hinv (nk,nk), tinv (nk,nk), tmat (nk,nk), tlag (nk,nk), ulag (nk,nk)	Real	Double precision matrices used in the semi-implicit solution procedure.
vecqq (nk), bam (nk), qvec (nk), alfcor (nk), dq (nk), alf (nk), evreal (nk)	Real	Double precision vectors used in the semi-implicit solution procedure.
sigma (nk)	Real	Array for calculating relaxation coefficients.
dqmqa5 (nk)	Real	Convective adjustment array.
wts (nkp1)	Real	Array used in semi-implicit solution procedure.
ds, dsi, dsi2, dssq	Real	The horizontal grid spacing; Δs, and related parameters; Δs^{-1}, $(2\ \Delta s)^{-1}$, Δs^{2}.
bet65,dtmax, alfnm	Real	Constants for semi-implicit scheme.

COMMON / THREED/

This common block contains all three-dimensional arrays.

Variable	Type	Description
rmm (nk,ni,nj), rm (nk,ni,nj), rmp (nk,ni,nj)	Real	Arrays for mixing ratio at previous, current and next time step.
um (nk,ni,nj), u (nk,ni,nj), up(nk,ni,nj)	Real	Arrays for *U*-component of velocity at previous, current, and next time step.
vm (nk,ni,nj), v (nk,ni,nj), vp (nk,ni,nj)	Real	Arrays for *V*-component of velocity at previous, current, and next time step.
phi(nk,ni,nj), omega(nk,ni,nj)	Real	Arrays for geopotential height and vertical velocity.
tm(nk,ni,nj), t(nk,ni,nj), tp(nk,ni,nj)	Real	Arrays for temperature at previous, current and next time step.
sigdot(nk,ni,nj), tdot(nk,ni,nj), qdot(nk,ni,nj)	Real	Arrays for $\dot{\sigma}$, $\dot{T}$, and $\dot{R}$.

COMMON / TWOD/

This common block contains all two-dimensional arrays.

Variable	Type	Description
trainm(ni,nj), train(ni,nj), trainp(ni,nj)	Real	Arrays for total rainfall at the previous, current, and next time step (convective + large scale).
tsm(ni,nj), ts(ni,nj), tsp(ni,nj)	Real	Arrays for soil temperature at the first soil layer at the previous, current and next time step.
tdm(ni,nj), td(ni,nj), tdp(ni,nj)	Real	Arrays for temperature of the second soil layer at the previous, current, and next time step.
wsm(ni,nj), ws(ni,nj), wsp(ni,nj)	Real	Arrays for soil moisture at the first soil layer at the previous, current, and next time step.
wdm(ni,nj), wd(ni,nj), wdp(ni,nj)	Real	Arrays for soil moisture at the second soil layer at the previous, current, and next time step.
zs(ni,nj), corp(ni,nj)	Real	Arrays of terrain height and Coriolis parameter.
em(ni,nj), emu(ni,nj), emv(ni,nj)	Real	Arrays of map factor at the temperature, and U and V (staggered) grid points.
albd(ni,nj), ruf(ni,nj), stmp(ni,nj), smos(ni,nj)	Real	Arrays for surface albedo, roughness length, sub-surface temperature and sub-surface moisture. All are specified in the pre-processor and input to CSC Maps.
dummy1(ni,nj), dummy2(ni,nj), dummy3(ni,nj), dummy4(ni,nj)	Real	Dummy arrays used for extra calculations in various subroutines.
qlat(ni,nj), qlong(ni,nj)	Real	Arrays for latitude and longitude of each grid point.
crain(ni,nj), lrain(ni,nj)	Real	Arrays for convective and large scale rainfall.
wgt(ni,nj), w2(ni,nj)	Real	Weights used for nesting scheme. w2=(1.0-wgt)
hlatn(ni,nj), hsens(ni,nj)	Real	Arrays for latent and sensible heat.

COMMON / ONEDR/

This common block contains real arrays and variables.

Variable	Type	Description
tbar (nk)	Real	Average temperature for each sigma level. Set in NESTIN.
q (nk)	Real	Sigma levels. Set in CONST.
rtbar (nk)	Real	Real variable for tbar.
tcool (nk)	Real	Newtonian cooling to space (set in PARAM).
qph (nk), qipk (nkp1)	Real	Half sigma levels, storage array
bet (nk), dtodq (nk), gama (nk)	Real	
stlat (nout), stlong (nout) stni (nout), stnj (nout)	Real	latitudes and longitudes and corresponding real grid point values of station variables.
istn (nout), jstn (nout)	Real	Integer grid coordinates of output stations.
philog (nout)	Real	Grid orientation at output stations.
proj (7)	Real	Array of projection parameters defining grid.
rdir (nij/3)	Real	Array of real numbers stored n the directory header of direct access input and output files.
tfilt, tfiltm	Real	Filter coefficients used for Asselin Filter.
soldec	Real	Solar declination angle.
hrtime	Real	Real time in hours.
rhf	Real	Relative humidity at which large scale rain triggers.
diffwt	Real	Horizontal diffusion coefficient (not used).
hours, drfact, dtop, ffac, rmdfwt	Real	Run time in hours, time step reduction factor, default top of atmosphere, filter factor, moisture diffusion coefficient, respectively.
precp, precta, cks, eke, pe, psbar, trhat, vromg	Real	Precipitation variables, map factor storage variable, kinetic and potential energy variables, respectively.
cp, g, r roncp	Real	Specific heat of dry air, gravitational constant, gas constant, R/Cp.
d1, d2, d3	Real	Depth of soil layers used in surface temperature and moisture prediction scheme.
ww1, ww2	Real	Soil temperature calculation constants for second layer.

COMMON / INTEGR/

This is a common block for integer arrays and variables.

Variable	Type	Description
land (ni,nj)	Integer	Land/sea mask (1 for land, 0 for sea).
itns (nk)	Integer	Dummy array for Helmholtz solver.
nstep	Integer	Integer timestep, initially zero in PARAMS and incremented by 1 at end of time-stepping loop.
nstep1, nstep2	Integer	First and last time steps.
irst	Integer	Restart time step.
nestep	Integer	Next time step at which nesting values are read.
nphys	Integer	Flag determining how frequently the physics routine is called.
nest	Integer	Flag for nesting scheme (0= open boundaries with radiation condition, 1= nesting in tendencies, 2= nesting in absolute values.
nsthrs, nhours, inthrs, nb, modes	Integer	Nesting time in hours, total run time in hours, nesting interval in hours, number of nesting grid points, resp.
idate, itime, kdate, ktime, nstime, ilog, logi, logj, iplot	Integer	Analysis date, analysis time, forecast date, forecast time, nesting time, resp.
nx, ny, nz	Integer	Actual model grid dimensions in the x, y, and z direction.
nxy, nzm1, nzm2, nzm3, nzm4, nzp1	Integer	Total number of horizontal grid points, number of sigma levels minus 1,2,3,4 and plus 1.
lphys, lsurf, lconls, lconcu, lverdf, lhdiff, lshcon	Integer	Flags for running physics scheme, the surface scheme, large scale rain, Kuo scheme, vertical diffusion, horizontal diffusion and shallow convection, resp.
nstns	Integer	Number of stations for time series output.
kdtop	Integer	Top level for application of vertical diffusion.
ilaps	Integer	Integer lapse time.
jproj	Integer	Flag representing the required map projection (3=Lambert conformal, 9= mercator).
idir (nij/3)	Integer	Array storing all integer information for the directory of a direct access file.

COMMON / ONEDC/

This is a common block for character arrays.

Variable	Type	Description
cdir (nij/3)	Char	Array storing all character information for the directory of a direct access file.
stname (nout)	Char	Array storing place names for time series output.

CSC GCOM2D COMMON Blocks

COMMON/ ARRAYS/

This is the main common block for GCOM2D. It carries all two-dimensional arrays.

Variable	Type	Description
twx (ni,nj), twy(ni,nj), tpa(ni,nj)	Real	Wind stress and surface pressure tendencies.
tz(ni,nj)	Real	Sea level nesting tendencies.
up (0:ni, 0:nj), vp (0:ni, 0:nj), zp (ni,nj)	Real	Velocity components and sea level perturbation at new time step.
uo(0:ni, 0:nj), vo(0:ni, 0:nj), zo(ni,nj)	Real	Velocity components and sea level perturbation at old time step.
ubar(0:ni, 0:nj), vbar(0:ni, 0:nj)	Real	U at V grid points and V at U grid points.
zres(ni,nj), zcg(ni,nj)	Real	Nesting arrays.
wspeed(ni,nj), wdir(ni,nj), pa(ni,nj)	Real	Wind speed, wind direction and atmospheric pressure.
wx(0:ni, 0:nj), wy(0:ni, 0:nj)	Real	Wind stress components.
pfz(ni,nj), pfu(0:ni, 0:nj), pfv (0:ni, 0:nj)	Real	Map factors at U, V and ζ grid points.
depz(ni,nj), depu(0:ni, 0:nj), depv(0:ni, 0:nj)	Real	Depth at U, V, and ζ grid points.
height(ni,nj), spfeat(ni,nj), topog2(ni,nj)	Real	Topography arrays.
cf(ni,nj), fv(ni,nj)	Real	Coefficient of bottom friction, Coriolis force.
tdhgt(ni,nj), e(ni,nj), wgt(ni,nj)	Real	Tidal height, total energy, and nesting weights, resp.
dummy0(ni,nj), dummy1(ni,nj), dummy2(ni,nj), dummy3(0:ni, 0:nj), dummy4(0:ni, 0:nj)	Real	Working arrays.
grav1(ni,nj)	Real	Storage array.
iminz (nj), imaxz(nj), iminu(0:ni, 0:nj), imaxu(0:ni, 0:nj), iminv(0:ni, 0:nj), imaxv(0:ni, 0:nj)	Integer	The first sea point on west of grid and last sea point on east of grid on each row at z, u, and v points.
jproj	Integer	Projection parameter flag.
proj (7)	Real	Projection parameter array.
llog	Logical	Logical unit of log file
depmin, stdxym, g, h0cbf	Real	Minimum bathymetric depth, standard grid spacing (m), acceleration due to gravity, default value of bottom friction, respectively.

CSC GCOM3D COMMON Blocks

COMMON/ THREED/
This is the main common block for GCOM3D. It carries all two-dimensional arrays.

Variable	Type	Description
wp(nk,ni,nj), wo(nk,ni,nj), wm(nk,ni,nj)	Real	Vertical velocity at next, present, and last time step.
rhop (nk,ni,nj)	Real	Density.
cevu(nk,ni,nj), cevv(nk,ni,nj)	Real	Coefficient of eddy viscosity at U and V points.

COMMON/ TWOD/
This is the main common block for GCOMD. It carries all two-dimensional arrays.

Variable	Type	Description
kmaxz (0:ni, 0:nj), kmaxu(0:ni, 0:nj), kmaxv 0:ni, 0:nj)	Real	Number of vertical levels at each Z, U, and V grid point.
bdepu(ni,nj), bdepv(ni,nj), bdepz(ni,nj)	Real	Thickness of bottom layer at U, V, and Z grid points.
h (nk)	Real	Thickness of each vertical layer.
depk (0:nk)	Real	Depth of each vertical layer.
dzbz(ni,nj), dzbu(ni,nj), dzbv(ni,nj)	Real	Distance between mid point of bottom layer and mid point of second bottom layer at Z, U, and V points.